TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

PETTERI KARJALAINEN

KANTAVIEN RAKENTEIDEN ALGORITMIAVUSTEISEN RAKENNESUUNNITTELUPROSESSIN KEHITTÄMINEN

Diplomityö

Tarkastaja:
professori Mikko Malaska
Tarkastaja ja aihe hyväksytty Talouden ja rakentamisen tiedekuntaneuvoston kokouksessa 27. elokuuta 2018

TIIVISTELMÄ

PETTERI KARJALAINEN: Kantavien rakenteiden algoritmiavusteisen rakennesuunnitteluprosessin kehittäminen
Tampereen teknillinen yliopisto
Diplomityö, 97 sivua
Lokakuu 2018
Rakennustekniikan diplomi-insinöörin tutkinto-ohjelma
Pääaine: Rakennesuunnittelu
Tarkastaja: professori Mikko Malaska

Avainsanat: algoritmiavusteinen suunnittelu, algoritmi, suunnitteluprosessi, Grasshopper, BIM, FEM

Tämän tutkimuksen tavoitteena on tutkia ja kehittää rakennuksen kantavien rakenteiden algoritmiavusteista rakennesuunnitteluprosessia. Tutkimusmenetelmänä käytetään kirjallisuustutkimusta sekä tapaustutkimusta. Tutkimus aloitetaan analysoimalla perinteistä, nykyään yleisesti käytettävää rakennesuunnitteluprosessia. Tämän jälkeen tarkastellaan algoritmiavusteisen suunnittelun ominaisuuksia ja arvioidaan niiden soveltuvuutta erilaisiin suunnittelutehtäviin. Tutkimuksessa tarkastellaan yksittäisten algoritmisten työkalujen lisäksi koko suunnitteluprosessin toimivuutta.

Algoritmiavusteista suunnitteluprosessia tutkitaan kuvitteellisen suunnittelukohteen avulla. Suunnittelussa keskitytään tarkastelemaan geometrisesti haastavaa teräsrakenteista kattorakennetta. Kattorakennetta ei pyritä mitoittamaan ja mallintamaan täydellisesti vaan tarkoituksena on keskittyä algoritmisen suunnitteluprosessin ja sen osatekijöiden toimivuuteen. Algoritmisen suunnittelun alustana toimii Rhinoceros 3D -ohjelman lisäosa, Grasshopper. Työssä toteutettava algoritminen suunnitteluprosessi etenee aina arkkitehdin algoritmipohjaisesti tuotetusta geometriasta toteutuskelpoisiin ja lujuudellisesti varmennettuihin rakenteisiin ja liitoksiin. Suunnittelussa linkitetään algoritmien avulla älykkäästi toisiinsa sekä arkkitehtimalli, laskentamalli että rakennemalli.

Tutkimuksen tulokset vastaavat melko hyvin kirjallisuudesta saatavia tuloksia. Algoritminen suunnitteluprosessi toimii kaikilta osiltaan suhteellisen hyvin ja sen vahvuudet ja ongelmakohdat ovat tunnistettavissa. Algoritmiset menetelmät mahdollistavat tietyissä rajoissa hyvin tehokkaan muutosten toteuttamisen. Suunnittelutiedon linkittyminen helpottaa suunnittelijoiden välistä yhteistyötä, vähentää päällekkäistä työtä ja tehostaa näin koko suunnitteluprosessia. Lisäksi algoritmit helpottavat ja nopeuttavat erityisesti geometrisesti haastavien sekä systemaattisesti toistuvien rakenteiden suunnittelua. Algoritmiavusteisilla menetelmillä on myös puutteensa. Jotta algoritmit toimisivat hyvin, tulee niitä luotaessa ottaa huomioon kaikki mahdolliset rakenteisiin kohdistuvat muutokset. Tämä on usein käytännössä mahdotonta. Lisäksi algoritmien avulla toteutettavassa reaaliaikaisessa pilvipohjaisessa tiedonsiirrossa tulee kiinnittää erityistä huomiota käytettävien työkalujen tiedonsiirtokapasiteetin riittävyyteen.

ABSTRACT

PETTERI KARJALAINEN: Development of algorithm-aided design process for structural engineering
Tampere University of Technology
Master of Science Thesis, 97 pages
October 2018
Master's Degree Programme in Civil Engineering
Major: Structural Engineering
Examiner: Professor Mikko Malaska

Keywords: algorithm-aided design, algorithm, design process, Grasshopper, BIM, FEM

The aim of this study is to examine and develop an algorithm-aided design process for structural engineering. Research methods to be used are a literary research and a case study. In the beginning of the study, a traditional and commonly used design process is being analysed. Subsequently, the features of algorithm-aided design are studied and their suitability for different design tasks is evaluated. In addition to the individual algorithmic tools, the study examines the functionality of the entire design process.

The algorithm-aided design process is being studied using an imaginary design case. The structural design process is being concentrated to the geometrically challenging roof structure made of steel. The purpose of the study is not to calculate and model the structure perfectly but to focus on the functionality of the algorithmic design process and its components. The algorithm-aided design is carried out using the Rhinoceros 3D software and its add-on Grasshopper. The algorithmic design process proceeds all the way from architect's algorithm-based geometry to feasible and structurally verified structures and joints. In the design process, the architectural model, the calculation model and the structural model are linked algorithmically with each other.

The results of the study are moderately equivalent to the results given in the literature. The algorithm-aided design process operates in all respects relatively well and its strengths and deficiencies can be identified. Algorithmic methods make it possible to implement very effective changes within certain limits. The intelligent linking of design data facilitates co-operation between designers, reduces overlapping work and makes the whole design process more effective. In particular, algorithms facilitate and accelerate the design of geometrically challenging and systematically repetitive structures. On the other hand, algorithm-aided methods also have their deficiencies. In order for the algorithms to work appropriately, all the possible changes to the generated structures should be considered, when the algorithms are created. This is often virtually impossible. In addition, a cloud-based data transfer with algorithms is rapidly encountering a data transfer maxim, which makes the implementation of data transmission much more difficult.

ALKUSANAT

Tämä tutkimus tehtiin osana Tampereen teknillisen yliopiston maisterivaiheen opintoja ja toteutettiin A-Insinöörit Suunnittelu Oy:n toimeksiannosta. Haluan kiittää professori Mikko Malaskaa työn tarkastajana toimimisesta. Kiitokset kuuluvat myös Ilari Pirhoselle työn ohjaamisesta sekä muille työkavereille mielenkiinnosta työtäni kohtaan. Lisäksi haluan kiittää opiskelukavereitani sekä perhettäni tuesta diplomityöprojektin sekä koko yliopisto-opintojen ajalta.

Opinnot ovat nyt takanapäin ja koen kehittyneeni niiden aikana paljon ammatillisesti sekä yleisesti ihmisenä. Erityisesti läheiset opiskelukaverini kulkivat kanssani koko opiskeluajan ja tekivät siitä ikimuistoisen kokemuksen. Vaikka opintojen loppuminen tuntuu hieman haikealta, jatkuu elämänmittainen oppiminen myös tulevaisuudessa. Ja onnekseni voin jatkaa myös uutta taivalta välittävän perheen ja hyvien ystävien seurassa.

Helsingissä, 19.9.2018

Petteri Karjalainen

SISÄLLYSLUETTELO

1. JOHDANTO

Rakennusalan suhdanteet ovat tällä hetkellä korkealla ja vuosittaiset rakennusmäärät suuria. Tämä tarkoittaa sitä, että myös rakennesuunnittelua harjoittavilla yrityksillä on runsaasti töitä. Rakennushankkeiden aikataulut ovat tyypillisesti hyvin kireitä ja usein jopa epärealistisia (Lassila 2016). Kun suunnittelun aikataulusta tehdään kireä, vaikuttaa se usein suunnittelun laatuun negatiivisesti. Suunnittelun laatu sitä vastoin korreloi rakennushankkeen onnistumiseen ja näin ollen vaikuttaa suoraan hankkeen toteutettavuuteen ja kokonaiskustannuksiin (Naumanen 2015). Rakennusalalla kohdattavat ongelmat vaikuttavat hyvin laajasti väestöön ja erityisesti rakennusten terveellisyyteen ja turvallisuuteen liittyvät ongelmat puhututtavat. Puutteet ja heikkoudet on tunnistettu ja niiden merkittävänä ratkaisuna pidetään digitalisaation kehittymistä sekä uusia teknologisia ratkaisuja. Tämä tarkoittaa sitä, että mahdollisia ratkaisuja tulee tutkia, analysoida ja kehittää.

Kuten Sulankivi et al. (2002) tutkimuksessaan toteavat, suunnittelu-rakentamisprosessissa on monia kehittämistarpeita ja -mahdollisuuksia. Mahdollisuuksia on erityisesti rakennushankkeeseen osallistuvien osapuolien välisen yhteistyön ja kommunikoinnin kehittämisessä. Tämä pätee myös nykyään yleisesti käytettävään rakennesuunnitteluprosessiin. Rakennesuunnitteluprosessin sisällä sekä rakennesuunnittelijan ja muiden hankeosapuolien välillä tapahtuvassa tiedonsiirrossa on monia mahdollisia kehittämiskohteita. Haasteena on ennen kaikkea kehittää teknisiä ratkaisuja, joiden avulla eri osapuolien välistä yhteistyötä voidaan parantaa. Nykyään käytännössä kaikki rakennesuunnittelutehtävät toteutetaan tietokoneavusteisesti. Monet suunnitteluohjelmat helpottavat suunnittelijoiden työtä huomattavasti, mutta samalla ne luovat myös täysin uusia työtehtäviä. Yksi suuri haaste on monien erilaisten suunnitteluohjelmistojen käyttäminen ja erityisesti niistä saatavan suunnittelutiedon yhdistäminen.

Suunnittelun laadun ja rakennushankkeen onnistumisen välillä voidaan katsoa olevan useita korrelaatioita. Suunnittelulla voidaan vaikuttaa toteutettavuuden ja kokonaiskustannusten lisäksi esimerkiksi tilojen käytettävyyteen ja estetiikkaan. (Naumanen 2015) Nykyisessä suunnitteluprosessissa edellä esitettyihin tekijöihin voi vaikuttaa pääasiassa vain arkkitehti. Myös rakennesuunnittelija pystyy vaikuttamaan valinnoillaan esimerkiksi hankkeen kustannuksiin, mutta sen vaikuttavuus on nykyisellään vähäinen. Algoritmiavusteisilla suunnittelumenetelmillä rakennesuunnittelija voi kuitenkin tehdä helpommin yhteistyötä arkkitehdin kanssa ja näin osallistua suunnitteluun heti rakennushankkeen alkuvaiheessa. Tällöin rakennesuunnittelijan ammattitaitoa voidaan hyödyntää huomattavasti paremmin alustavassa suunnittelussa ja näin saavuttaa merkittäviä kustannussäästöjä.

Algoritmiavusteisella suunnitteluprosessilla uskotaan saavutettavan monia etuja perinteiseen suunnitteluprosessiin verrattuna. Tässä tutkimuksessa tarkastellaan koko rakennesuunnittelun tehtäväkenttää ja tutkitaan algoritmiavusteisten menetelmien soveltuvuutta niihin. Tavoitteena on saavuttaa algoritminen suunnitteluprosessi, jossa edetään aina arkkitehdin algoritmiavusteisesti muodostamasta geometriasta toteutuskelpoisiin ja lujuudellisesti varmennettuihin rakenteisiin ja liitoksiin. Tutkimuksessa keskitytään tarkastelemaan geometrisesti haastavan, teräsrakenteisen kattorakenteen suunnitteluprosessia. Tässä luvussa tarkastellaan vielä syvemmin tutkimuksen taustaa, tavoitteita ja rajausta. Lisäksi käydään läpi tutkimuksen suoritus sekä tutkimusraportin rakenne.

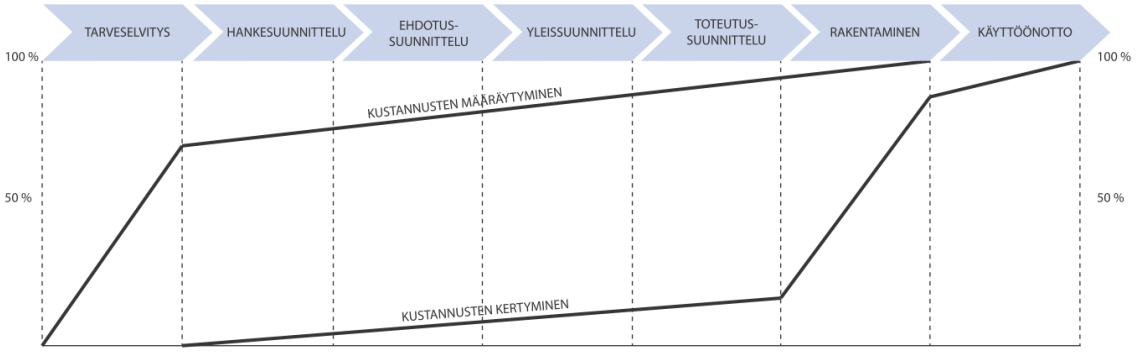
1.1 Tutkimuksen tausta

Rakennesuunnittelussa käytetään tällä hetkellä hyvin vähän algoritmiavusteisia suunnittelumenetelmiä. Rakenteiden analysointi, mitoitus ja mallinnus tapahtuvat eri ohjelmistojen avulla, joissa algoritmipohjaisia suunnittelumenetelmiä hyödynnetään vielä hyvin rajoittuneesti. Rakennusalalle soveltuvia algoritmiavusteisia suunnitteluohjelmistoja löytyy markkinoilta jo useampia ja esimerkiksi arkkitehdit hyödyntävät niitä suunnittelussa selvästi rakennesuunnittelijoita enemmän. Algoritmiavusteisten suunnittelumenetelmien potentiaali on huomattu myös rakennesuunnittelijoiden keskuudessa ja niiden hyödyntämismahdollisuuksia on alettu kartoittaa myös rakennesuunnittelun näkökulmasta.

Rakennusten suunnittelu on iteratiivinen prosessi, jossa suunnitteluratkaisut muuttuvat ja tarkentuvat useita kertoja hankkeen edetessä. Suunnitteluun osallistuu useita eri tahoja, jotka kaikki vastaavat oman vastuualueensa suunnitelmista. Kun kaikkien osapuolien suunnitelmat vaikuttavat eriasteisesti toisiinsa ja suunnittelua tehdään osittain samanaikaisesti, muodostuu suunnittelusta monisyinen kokonaisuus. Tämän vuoksi suunnittelussa tarvitaan useita iteraatiokierroksia, joiden avulla päästään kaikkien suunnittelutahojen vaatimukset täyttäviin ratkaisuihin. Suunnitteluprosessi on iteratiivinen myös eri suunnittelutahojen sisällä. Esimerkiksi rakennesuunnittelija joutuu työssään usein kokeilemaan monia erilaisia rakennevaihtoehtoja ennen kuin lopullinen toteutettava rakenne varmistuu. Edellä kuvatut rakennusten suunnittelulle ominaiset iteratiiviset prosessit tarkoittavat sitä, että suurin osa suunnittelijoiden käyttämästä työajasta menee rakenteiden ja niiden suunnitelmien muokkaamiseen.

Muista suunnitteluosapuolista eniten rakennesuunnittelijan työhön vaikuttaa arkkitehti. Rakennesuunnittelija saa arkkitehdilta perinteisesti oman työnsä lähtötiedoiksi esimerkiksi IFC-mallin ja 2D-piirustuksia. Nykyisillä toimintatavoilla arkkitehti suunnittelee rakennuksen rungon dimensioiltaan hyvin pitkälle ennen kuin rakennesuunnittelija liittyy mukaan suunnitteluun. Rakennesuunnittelijan tehtävänä on ennen kaikkea varmistaa arkkitehdin runkoratkaisujen toimivuus ja mitoittaa rakenneosat ja niiden liitokset. Rakennejärjestelmä ei ole aina optimaalisin rakenteiden toimivuuden ja siten myös rakennuskustannusten kannalta, mutta arkkitehdin ja rakennesuunnittelijan yhteistyöllä yhteiset

tavoitteet yleensä saavutetaan. Kuten kuvasta 1 nähdään, määräytyvät rakennuskustannukset pääasiallisesti rakennushankkeen alkuvaiheessa. Tämä tarkoittaa sitä, että arkkitehdin hankkeen alkuvaiheessa tekemillä valinnoilla on huomattava vaikutus rakennushankkeen kokonaiskustannuksiin.



Kuva 1. *Kustannusten määräytymien ja kertyminen rakennushankkeen eri vaiheissa (RT 10-11226 2016, s. 1).*

Algoritmiavusteiset suunnittelumenetelmät soveltuvat erityisesti haastavien muotojen sekä toistuvien rakenteiden suunnitteluun. Niiden avulla myös muutosten tekeminen suunnitelmiin on huomattavasti perinteisiä suunnittelumenetelmiä helpompaa ja tehokkaampaa. Algoritmiavusteisella suunnittelulla onkin siis paljon potentiaalisia hyödyntämismahdollisuuksia koko rakennusalalla. Algoritmipohjaisen suunnittelun avulla iteratiivisia suunnitteluprosesseja pystytään mahdollisesti tehostamaan, minkä avulla pystytään vaikuttamaan myös rakennushankkeen kokonaiskustannuksiin positiivisesti. Kun suunnitteluprosessista tulee joustavampi muutoksille, voidaan myös kustannusten määräytymistä siirtää lähemmäs rakentamisvaihetta.

1.2 Tutkimuksen tavoitteet ja rajaukset

Tämän tutkimuksen päätavoitteena on tutkia ja kehittää rakennuksen kantavien rakenteiden algoritmiavusteista rakennesuunnitteluprosessia rakennesuunnitteluyrityksen näkökulmasta. Alatavoitteet, joiden avulla päätavoitteeseen pyritään, ovat:

1. Arkkitehdin algoritmipohjaisten suunnitelmien hyödyntäminen rakennesuunnittelun lähtötietona.
2. Algoritmipohjaisen mitoitus- ja mallinnusprosessin tutkiminen.
3. Liitosten mitoitus ja mallinnus algoritmisten suunnittelumenetelmien avulla.

Tutkimuksen teollisena tavoitteena on tutkia ja kehittää algoritmiavusteiseen rakennesuunnitteluun soveltuvaa suunnitteluprosessia sekä luoda algoritmipohjaisia työkaluja rakennesuunnittelun apuvälineeksi. Tiedollisena tavoitteena on algoritmiavusteisen suun-

nitteluprosessin kuvaaminen sekä sen hyötyjen, ongelmakohtien ja kehitystarpeiden tarkastelu. Algoritmiavusteisen suunnitteluprosessin toimivuutta verrataan myös perinteiseen, nykyisin yleisesti käytettävään, suunnitteluprosessiin.

Tutkimuksessa suunnitellaan kuvitteellinen rakennuskohde siten, että sekä arkkitehti että rakennesuunnittelija hyödyntävät algoritmiavusteisia suunnittelumenetelmiä. Tutkimuksessa keskitytään tarkastelemaan geometrisesti haastavaa teräsrakenteista kattorakennetta. Kattorakennetta ei pyritä mitoittamaan ja mallintamaan täydellisesti vaan tarkoituksena on keskittyä algoritmisen suunnitteluprosessin ja sen osatekijöiden toimivuuteen. Algoritmisen suunnittelun alustana toimii Rhinoceros 3D -ohjelman lisäosa, Grasshopper.

Työssä tarkastellaan algoritmiavusteista suunnitteluprosessia kolmen eri osatekijän avulla. Ensimmäinen tekijä on arkkitehdin ja rakennesuunnittelijan algoritmipohjaisten suunnittelumenetelmien yhteensovittaminen. Tämä koostuu arkkitehdin ja rakennesuunnittelijan algoritmisten suunnitteluprosessien integroimisesta sekä suunnittelutiedon linkittämisestä. Toinen tekijä on rakennesuunnittelussa hyödynnettävien algoritmiavusteisten mitoitus- ja mallinnusprosessien kehittäminen. Työssä tutkitaan, miten suunnitteluprosessi etenee ja minkälaisia työkaluja sen toteuttamiseksi tarvitaan. Suunnittelun ominaisuuksia tutkitaan monimuotoisen teräsrakenteisen kattorakenteen avulla. Kolmas työssä tarkasteltava osatekijä on liitoksen algoritminen mitoitus ja mallinnus. Tässä osassa on tarkoitus tutkia liitoksen mitoitusprosessia ja hyödyntää laskentamallista saatavia kuormitustietoja liitoksen parametrisessa mallintamisessa.

1.3 Tutkimuksen suoritus ja rakenne

Tutkimus koostuu karkeasti kahdesta osasta, joissa käytetään tutkimusmenetelmänä sekä kirjallisuus- että tapaustutkimusta. Työssä ensin tarkasteltava teoriaosa koostuu pääasiallisesti kirjallisuusanalyysistä, jonka avulla perehdytään algoritmiavusteisen suunnittelun ominaisuuksiin sekä verrataan sitä perinteisiin suunnittelumenetelmiin. Aineistoa kerätään pääasiassa alan kirjallisuudesta sekä tieteellisistä artikkeleista. Kirjallisuusanalyysin pohjalta opittua taustatietoa hyödynnetään tämän jälkeen tutkimusosan suorittamisessa. Siinä tutkimusmenetelmänä käytetään pääasiassa tapaustutkimusta eli niin sanottua casetutkimusta. Tapaustutkimuksen avulla tutkitaan algoritmiavusteista suunnitteluprosessia, kehitetään algoritmisia työkaluja sekä arvioidaan niiden toimivuutta, hyötyjä ja haittoja.

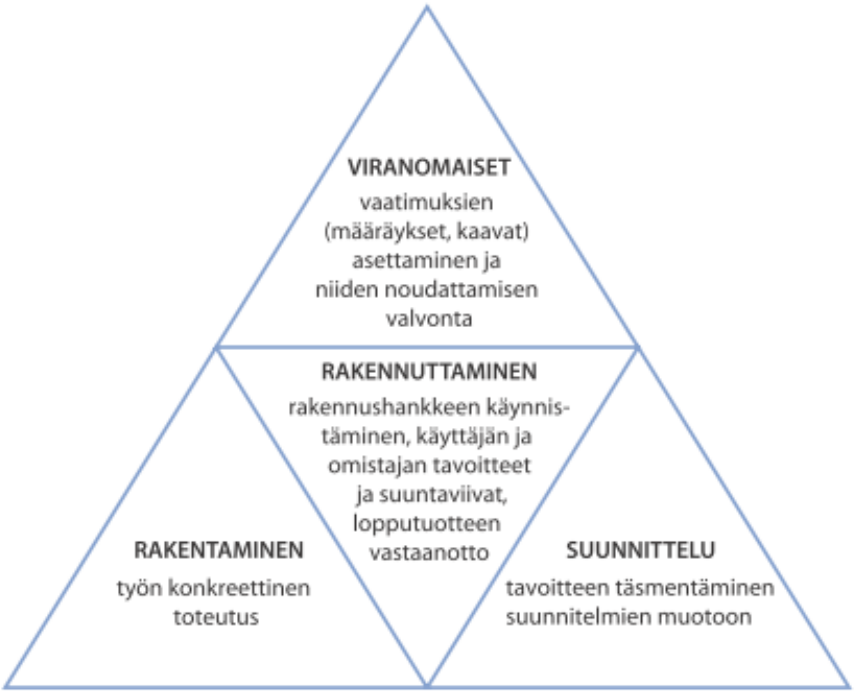
Tässä työssä selvitetään, miten kantavien rakenteiden algoritmiavusteinen suunnitteluprosessi etenee ja mitkä ovat sen hyödyt, haitat ja jatkokehitystarpeet. Lisäksi tarkastellaan algoritmiavusteisen ja perinteisen suunnitteluprosessin eroja. Tutkimus tuottaa myös algoritmiavusteisia työkaluja rakennesuunnittelua harjoittavan yrityksen käytettäväksi. Tutkimuksessa keskitytään tarkastelemaan erityisesti arkkitehdin tietomallin sekä rakennesuunnittelijan tieto- ja laskentamallin linkittämiseen tarvittavia algoritmisia työkaluja. Koska käytännön työkaluja ja algoritmeja ei voida julkaista tämän työn viitekehyksessä,

pyritään algoritmien tarkoitus, toiminta ja käyttökohteet raportoimaan kirjallisessa muodossa.

Tämä tutkimus koostuu kuudesta luvusta. Ensimmäisessä luvussa käydään läpi tutkimuksen tausta, tavoitteet ja rajaukset sekä suoritus ja rakenne. Toinen luku on ensimmäinen osa työn teoriaosaa ja siinä tarkastellaan perinteistä, nykyään yleisesti käytettyä rakennesuunnitteluprosessia. Kolmas luku muodostaa toisen osan työn teoriaosuutta ja siinä tarkastellaan algoritmiavusteista suunnittelua. Neljännessä luvussa käydään läpi työssä toteutettava tapaustutkimus ja tutkitaan sen osakokonaisuuksia ja niiden ominaisuuksia. Tämän jälkeen viidennessä luvussa analysoidaan tutkimustuloksia ja käydään läpi tutkittujen asioiden vahvuuksia, puutteita ja jatkokehitystarpeita. Viimeisessä luvussa kerrataan vielä koko työn sisältö ja esitellään tutkimuksesta tehdyt johtopäätökset.

2. PERINTEINEN RAKENNESUUNNITTELUPROSESSI

Rakennushankkeiden muuttuessa yhä suuremmiksi, monimutkaisemmiksi ja vaikeammin hallittaviksi kokonaisuuksiksi rakennushankkeeseen osallistuvien osapuolien välisen yhteistyön tärkeys korostuu. Kun rakennuksen suunnitteluun ja toteutukseen osallistuu useita eri tahoja, on eri toimijoiden välisten vuorovaikutustaitojen hallitseminen välttämätöntä onnistuneen lopputuloksen kannalta. (Junnonen & Kärnä 2015) Kuten kuvasta 2 nähdään, rakennushankkeen osapuolet voidaan jakaa karkeasti neljään eri toimijaan: rakennuttamiseen, suunnitteluun, rakentamiseen sekä rakennushankkeen toteutusta valvovaan viranomaistahoon (RT 10-11222 2016, s. 1).

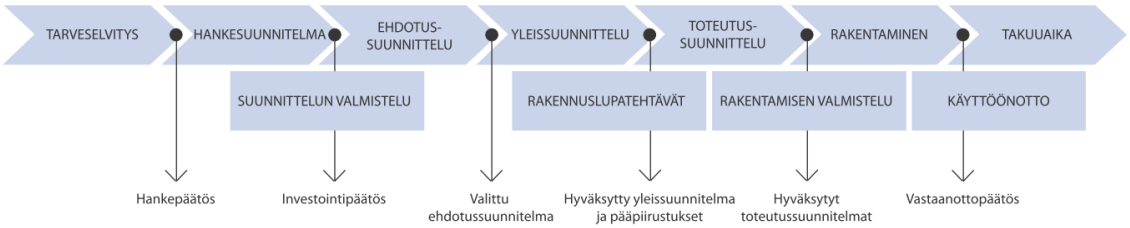


Kuva 2. *Rakennushankkeen osapuolet (RT 10-11222 2016, s. 1).*

Rakennusten suunnitteluun osallistuu tyypillisesti useita eri suunnittelualojen asiantuntijoita. Rakennushankkeiden muuttuessa monimutkaisemmiksi kokonaisuuksiksi monet suunnitteluyritykset ovat erikoistuneet jonkin tietyn kokonaisuuden suunnitteluun. Tämän seurauksena myös suunnittelusta on muodostunut monitahoinen kokonaisuus, jossa monet eri suunnittelualat työskentelevät vuorovaikutuksessa toistensa sekä muiden rakennushankkeen osapuolien kanssa. Eri rakennushankkeet sisältävät erilaisia suunnittelutehtäviä ja niiden painotukset vaihtelevat rakennushankkeen ominaispiirteistä riippuen. Koska rakennushankkeiden sisältö ja laajuus voivat vaihdella hyvinkin paljon, ei rakennusten suunnittelussa aina tarvita kaikkia eri suunnittelutahoja. Rakennesuunnittelu on

kuitenkin tyypillisesti osa suunnitteluprosessia, koska rakenteiden turvallisuus ja tekninen toimivuus on aina varmistettava. (RT 10-11222 2016)

Talonrakennushanke on useasta vaiheesta koostuva projekti. Yleisesti tunnetut vaiheet ovat: tarveselvitys, hankesuunnittelu, ehdotussuunnittelu, yleissuunnittelu, toteutussuunnittelu, rakentaminen ja käyttöönotto. Rakennushankkeen vaiheet on esitetty kuvassa 3.



Kuva 3. *Rakennushankkeen vaiheet (RT 10-11224 2016, s. 1).*

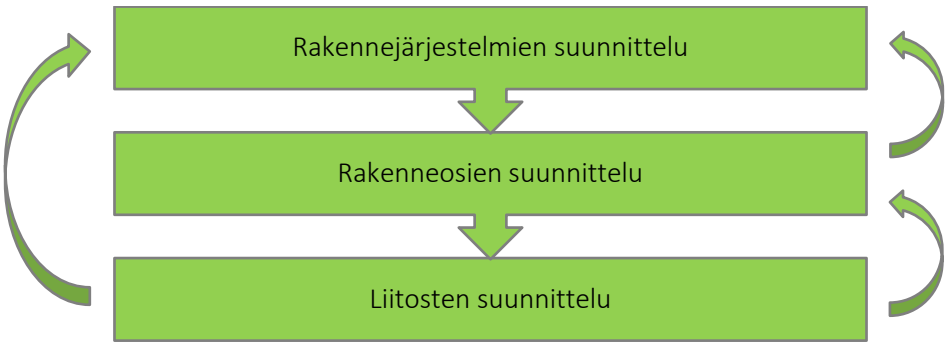
Rakennesuunnittelija osallistuu erityisesti yleis- ja toteutussuunnitteluun, mutta erilaisia työtehtäviä saattaa sisältyä myös muihin projektin vaiheisiin. Rakennesuunnittelijan tärkein tehtävä on varmistaa rakennuksen rakennetekninen toimivuus. Tämä tarkoittaa käytännössä perustus- ja runkoratkaisujen kehittämistä, rakenteiden mitoitusta sekä rakennuksen toteutettavuuden ja rakennusfysikaalisen toimivuuden varmistamista. Tieto suunnitelluista rakenteista tulee välittää myös muille rakennushankkeen osapuolille, mikä tapahtuu nykyään pääasiallisesti kolmiulotteisen tietomallin sekä siitä tuotettujen piirustusten avulla. (RT 10-11222 2016; RT 10-11224 2016)

Rakennesuunnitteluprosessin voi karkeasti jakaa kolmella tasolla tapahtuvaan suunnitteluprosessiin. Ensimmäinen ja laajin taso sisältää koko rakennushankkeen aikaisen suunnitteluprosessin ja sisältää täten kaiken hankkeen aikana tapahtuvan rakennesuunnittelun. Suunnittelun sisältö koostuu hankkeen edetessä tarkentuvista suunnitelmista, jotka toteutetaan yhteistyössä muiden hankeosapuolien kanssa. Hankkeen edetessä rakennesuunnittelija saa esimerkiksi lähtötietoja arkkitehdilta ja raportoi omia suunnitelmiaan talotekniikkasuunnittelijoille. Tämän prosessin voi katsoa koostuvan kolmesta osasta: rakennejärjestelmien suunnittelu, rakenneosien suunnittelu ja osien välisten liitosten suunnittelu. Kolmea edellä esitettyä suunnittelukokonaisuutta voidaan pitää suunnittelun toisena prosessitasona. Nykyajan tietomallinnettavissa hankkeissa kaikki edellä esitetyt kokonaisuudet voidaan jakaa edelleen kolmeen osaan. Esimerkiksi rakenneosien suunnittelu koostuu rakenteiden mitoituksesta, mallinnuksesta ja suunniteltujen rakenteiden raportoinnista. Näitä voidaan pitää viimeisenä ja suunnittelukokonaisuuden kannalta suppeimpana prosessitasona. Näiden prosessien sisältö eroaa jo huomattavasti toisistaan, mutta niistä on silti löydettävissä tyypilliset osatekijät, joilla tavoiteltu tulos saavutetaan.

Tässä luvussa tarkastellaan perinteistä rakennesuunnitteluprosessia erityisesti rakennuksen kantavien rakenteiden osalta. Perinteisellä suunnittelulla tarkoitetaan tässä tutkimuksessa nykyään yleisesti käytettyä tietomallintavaa rakennesuunnitteluprosessia. Luvussa keskitytään rakennesuunnittelun vaiheiden ja rakennesuunnittelijan tyypillisten tehtävien kuvaamiseen sekä arkkitehdin ja rakennesuunnittelijan välisen vuorovaikutuksen tutkimiseen. Todellisuudessa rakennesuunnittelijan sidosryhmiin kuuluvat läheisesti myös esimerkiksi tilaaja, urakoitsija sekä talotekniikan suunnittelijat, mutta tässä tutkimuksessa keskitytään ainoastaan arkkitehdin ja rakennesuunnittelijan väliseen rajapintaan. Lisäksi tarkastellaan rakennesuunnittelun vaikutuksia rakennushankkeen kokonaiskustannuksiin sekä nykyisen suunnitteluprosessin puutteita ja ongelmakohtia.

2.1 Rakennesuunnitteluprosessin vaiheet

Rakennesuunnittelu on monivaiheinen iteratiivinen prosessi, joka etenee rakennejärjestelmän alustavista selvityksistä kohti lopullisia toteutussuunnitelmia. Rakennesuunnittelu aloitetaan usein hyvin alustavista luonnossuunnitelmista, jotka projektin edetessä tarkentuvat vähitellen toteutettavien rakenteiden suunnitelmiksi. Monet rakennushankkeet ovat suuria kokonaisuuksia, jotka koostuvat useista rakennuksista, rakennejärjestelmistä ja rakenneosista. Kuten kuvasta 4 nähdään, rakennesuunnittelu voidaan jakaa karkeasti kolmeen osaan: rakennejärjestelmien, rakenneosien sekä rakenneosien välisten liitosten suunnitteluun. Kaikki nämä tekijät on otettava huomioon koko suunnitteluprosessin aikana, mutta pääasiallisesti suunnittelu etenee laajemmista kokonaisuuksista kohti tarkkoja detaljeja. Erilaisten rakennushankkeiden suunnittelukokonaisuudet saattavat erota hyvinkin paljon toisistaan, minkä vuoksi yksiselitteisten suunnitteluvaiheiden määrittäminen on vaikeaa.



Kuva 4. *Rakennesuunnittelun vaiheet.*

Rakennesuunnittelu voidaan jaotella sekä rakennushankkeen eri vaiheissa että yksittäisen suunnittelukokonaisuuden sisällä tapahtuviksi rakennesuunnittelijan vastuunalaisiksi tehtäviksi. Suomessa rakennusalan ohjeistuksia teettävä Rakennustieto Oy on koonnut vuonna 2013 rakennesuunnittelun tehtäväluettelosta ohjeen RT 10-11128. Luettelossa käydään läpi tavanomaisen talonrakennushankkeen rakennesuunnittelun tehtävät ja niiden tulokset. Tehtävät on jäsennelty kuvassa 3 esitettyjen rakennushankkeen vaiheiden

mukaan, minkä avulla nähdään rakennesuunnittelun eri vaiheet rakennushankkeen edetessä.

Vaikka rakennesuunnittelu tapahtuu pääasiallisesti ehdotus-, yleis- ja toteutussuunnitteluvaiheissa, saattaa rakennesuunnittelija osallistua hankkeeseen jo tarveselvitystä tehtäessä. Tarveselvityksessä tarkastellaan tarvittavien tilojen määrää ja laatua sekä tutkitaan erilaisten tilaratkaisujen edullisuutta. Rakennesuunnittelijan tehtävänä on pääasiassa tehdä alustavia rakennusteknisiä selvityksiä sekä avustaa rakentamisaikataulun ja kustannusennusteen laadinnassa. Rakennesuunnittelijan tehtävänä voi olla esimerkiksi arvioida erilaisten rakennevaihtoehtojen käyttöikää, muunneltavuutta sekä ympäristövaikutuksia. (RT 10-11128 2013, s. 2)

Hankesuunnittelussa tarkennetaan muun muassa rakennushankkeen laajuus-, aikataulu-, laatu- ja kustannustavoitteita. Myös rakennesuunnittelijan tehtävänä on tarkentaa tarveselvityksessä tehtyjä raportteja. Tavoitteena on huolehtia tarvittavien selvitysten laatimisesta sekä asettaa tavoitteet hankkeen suunnittelulle. Rakennesuunnittelijan tulee esimerkiksi arvioida rakennuspaikan rakennettavuutta ja hankkia muita lähtötietoja rakennettavuuden selvittämiseksi. Hankesuunnittelun jälkeen rakennushankkeessa tapahtuu suunnittelun valmistelu, jonka tuloksena syntyy suunnittelupäätös ja rakennesuunnittelu voidaan aloittaa. Suunnittelun valmistelussa kilpailutetaan ja valitaan suunnittelijat sekä tehdään suunnittelusopimukset. Lisäksi rakennesuunnittelija selvittää yhdessä tilaajan kanssa suunnittelutehtävän laajuuden, vaativuuden ja tarvittavat lähtötiedot. (RT 10-11128 2013, s. 3–4)

Kun suunnittelupäätös on saatu tehtyä alkaa ehdotussuunnittelu. Tämä on tyypillisesti hankkeen ensimmäinen vaihe, johon rakennesuunnittelija osallistuu aktiivisesti suunnittelusopimuksessa esitettyjen ehtojen mukaan. Ehdotussuunnittelussa vertaillaan vaihtoehtoisia suunnitteluratkaisuja ja valitaan niistä taloudellisesti ja toiminnallisesti toimivin ratkaisu. Rakennesuunnittelijan tehtävänä on myös varmistaa, että valittava ratkaisu täyttää suunnittelulle ja koko hankkeelle asetetut vaatimukset. Rakennesuunnittelijan vastuulla olevia tehtäviä ehdotussuunnittelussa ovat esimerkiksi:

- Määritellä alustavat kuormitukset, materiaalit ja käyttöikätavoitteet.
- Sopia tietomallipohjaisen suunnittelun sisältö ja laajuus.
- Tarkistaa omien suunnitelmien yhteensopivuus muiden suunnittelualojen suunnitelmien kanssa.
- Tarkastella valittuun ehdotusratkaisuun sopivat rakennejärjestelmävaihtoehdot.
- Laatia ehdotus käytettävistä rakennetyypeistä ja muista tyyppiratkaisuista.

Lopuksi ehdotussuunnitelmille hankitaan kirjallinen hyväksyntä ja valitaan lopullinen suunnitteluratkaisu yleissuunnittelun pohjaksi. (RT 10-11128 2013, s. 5–6)

Ehdotussuunnittelun jälkeen alkaa hankkeen yleissuunnittelu. Yleissuunnittelussa valitusta ehdotussuunnitelmasta aletaan kehittää toteutuskelpoista suunnitelmaa. Suunnitelmista tulee selvitä rakennuksen ja rakenneosien laajuus, määrät ja työtavat, jotta hankkeen toteutuskustannuksia voidaan tarkentaa. Myös yleissuunnitteluvaiheessa rakennesuunnittelijan tulee tarkistaa, että suunnitelmat vastaavat ennalta määritettyjä tavoitteita ja vaatimuksia. Yleissuunnitteluvaiheessa rakennesuunnittelun vastuunalaiset tehtävät lisääntyvät. Esimerkkejä rakennesuunnittelijan tehtävistä ovat:

- Täsmentää kuormitukset, materiaalit ja käyttöikätavoitteet.
- Tarkentaa tietomallipohjaisen suunnittelun sisältö ja laajuus.
- Laatia perustus- ja runkorakenteiden yleissuunnitelmat ja rakenneosaluettelot.
- Suunnitella käytettävät rakennetyypit ja varmistaa niiden rakennusfysikaalinen toimivuus.
- Laatia perustus- ja runkorakenteiden sekä liitosten periaatteelliset rakennelaskelmat.

Yleissuunnittelun tuloksena saadaan pääpiirustukset, jotka toimivat pohjana rakennuslupatehtävissä. Rakennuslupatehtävissä tarkistetaan ja täydennetään rakennesuunnitelmat ja suunnitelmiin liittyvät asiakirjat, jotta voidaan laatia rakennusvalvontaviranomaisen ohjeiden mukainen rakennuslupahakemus. (RT 10-11128 2013, s. 7–10)

Seuraavana rakennushankkeessa on vuorossa toteutussuunnittelu. Se on rakennesuunnittelun kannalta hankkeen tärkein vaihe, koska silloin kaikki suunnitelma-asiakirjat saavat lopullisen muotonsa. Toteutussuunnittelussa suunnitelmat tarkentuvat muotoon, jonka avulla voidaan määrittää rakennusosien määrät, työtavat ja laatutaso. Toteutussuunnittelu voidaan jakaa kahteen vaiheeseen: hankintoja ja toteutusta palvelevaan suunnitteluun. Rakenneosista tehdään hankintakyselyjä varten tuoteosamäärittelyt, jotta toteutettavat rakenteet vastaavat tarkasti suunnitelmia. Lisäksi toteutussuunnittelussa luodaan toteutuksen kannalta välttämättömät asennus- ja detaljisuunnitelmat. Rakennesuunnittelijalla on toteutussuunnittelussa monia tehtäviä, kuten esimerkiksi:

- Määritellä suunnittelurajat ja avustaa urakkarajaliitteen laatimisessa.
- Varmistaa kantavien rakenteiden mitoitus.
- Täydentää liitos-, asennus- ja liittymädetaljit.
- Laatia kaikkien talo-osien toteutussuunnitelmat ja hankinta-asiakirjat.
- Määritellä tuote- ja järjestelmäosien vaatimukset ja urakkarajat.
- Koota lopulliset rakennepiirustukset ja varmistaa tietomallin oikeellisuus.

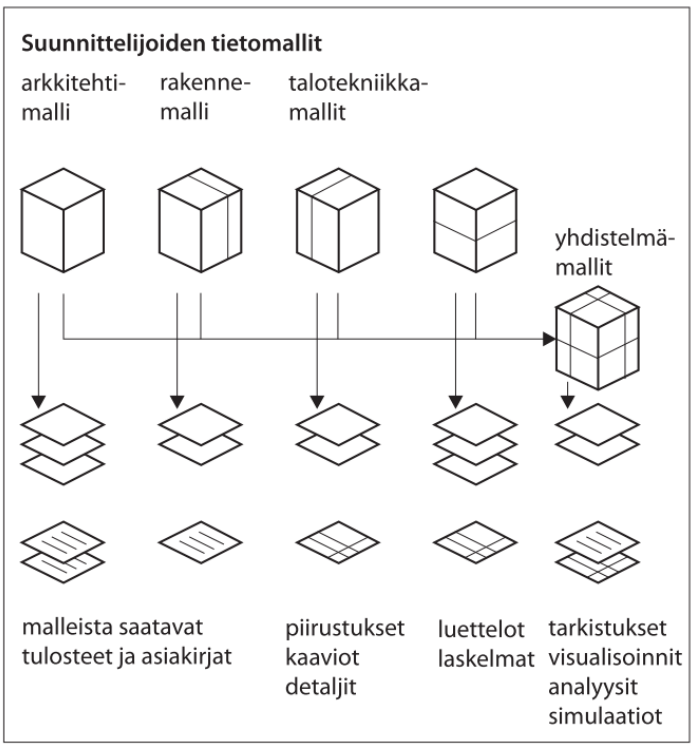
Toteutussuunnitelmien valmistuttua varmistetaan vielä eri suunnittelualojen suunnitelmien yhteensopivuus pääsuunnittelijan johdolla. Hyväksyttyjen toteutussuunnitelmien avulla kohteen rakentaminen voidaan aloittaa. (RT 10-11128 2013, s. 11–24)

Vaikka rakennushankkeessa siirrytään rakentamisvaiheeseen, riittää rakennesuunnittelijalla usein vielä tehtävää. Rakennesuunnittelija vastaa rakentamisen aikaisten muutosten

suunnittelusta ja niiden viranomaishyväksynnän hakemisesta. Lisäksi rakennesuunnittelija vastaa sovittujen suunnittelukokonaisuuksien rakentamisen valvonnasta sekä vastaa rakentamiseen liittyviin kysymyksiin. Suunnittelija osallistuu esimerkiksi työmaan aloitus- ja seurantakokouksiin sekä raudoitustarkastuksiin. Kun rakentaminen saadaan päätökseen, voidaan valmis rakennus ottaa käyttöön. Rakennesuunnittelijan tehtävänä on toimittaa lopulliset suunnitelma-asiakirjat ja tietomallit tilaajalle sekä tarvittaessa ohjeistaa käyttäjää rakennuksen käyttöön ja huoltoon liittyvissä kysymyksissä. (RT 10-11128 2013, s. 26–27)

2.2 Arkkitehtisuunnitelmat rakennesuunnittelun lähtötietona

Nykyajan rakennushankkeessa rakennesuunnittelu on riippuvainen monesta muusta suunnittelu- ja toteutustahosta. Tässä tutkimuksessa keskitytään kuitenkin tarkastelemaan ainoastaan rakennesuunnittelun ja arkkitehtisuunnittelun vuorovaikutusta. Arkkitehti vaikuttaa usein eniten rakennesuunnitteluun tarjoamalla lähtötiedot rakennesuunnittelun pohjaksi. Suunnittelu tapahtuu nykyään pääasiallisesti tietomallipohjaisesti ja tiedonsiirto eri osapuolien välillä tapahtuu IFC-tiedonsiirtoformaatin (Industry Foundation Class) avulla. Piirustuksia ja muita suunnitteluasiakirjoja käytetään yhä 3D-mallien rinnalla, mutta nekin tuotetaan yleensä tietomallien avulla. Kuvassa 5 on esitetty, kuinka eri suunnittelualojen mallit voidaan koota myös yhdeksi yhdistelmämalliksi. Yhdistelmämallin avulla voidaan tarkistaa suunnitelmien ristiriidattomuus sekä analysoida suunnittelukokonaisuutta.



Kuva 5. *Eri suunnitteluosapuolien tuottamat tietomallit ja niistä saatavat dokumentit (RT 10-10992, s. 5).*

Tällä hetkellä rakennesuunnittelu saa suunnittelun lähtötiedoksi arkkitehdilta IFC-tiedostona arkkitehtimallin sekä tarvittavat 2D-kuvannot. Hankkeen alkuvaiheessa sovitaan yhteiset tiedonsiirto- ja tiedonhallintaohjeet, joiden pohjalta määritellään esimerkiksi, kumpi edellä mainituista tiedoista on pätevä mahdollisten ristiriitaisuuksien ilmetessä. Yleensä ristiriitaisen tiedon ilmetessä käytettään rakennesuunnittelun lähtötietona piirustuksista löytyvää informaatiota. Tämä on kuitenkin muuttumassa tietomallinnuksen yleistyessä ja kehittyessä siihen, että tietomalli on jatkossa ensisijainen tiedonlähde. Koska arkkitehti tuottaa 2D-kuvat pääasiassa tietomallista, saatetaan tulevaisuudessa joidenkin piirustusten tuottamisesta luopua kokonaan.

Rakennushankkeen osapuolien välinen toimiva tiedonsiirto tehostaa rakennuksen suunnittelua. Tiedonsiirtoympäristö koostuu kolmesta peruskomponentista, jotka ovat sähköisessä muodossa oleva tieto, kaikkien osapuolien käytettävissä oleva tietovarasto sekä hankkeen eri osapuolet ja tietovaraston yhdistävä tietoverkko (Sulankivi et al. 2002, s. 23). Rakennushankkeen aikana muodostuu paljon erilaista dataa, jota on tärkeää koota järjestelmällisesti tietovarastoon. Tietomallinnuksen kehittyessä malleista saadaan tuotettua entistä enemmän dataa, mikä lisää tiedonsiirtoa entisestään.

Nykyään yleisesti käytetty suunnittelutiedon kokoamistapa on projektipankki. Projektipankki on usein ulkopuolisen palveluntarjoajan ylläpitämä tietovarasto, johon eri suunnitteluosapuolet tallentavat suunnitelmiaan. Suunnitelmien muuttuessa suunnittelijat päivittävät aina uusimmat suunnitelmansa projektipankkiin muiden osapuolien nähtäväksi. Täten projektin eri osapuolilla on aina käytössään uusimmat suunnitelmat. Arkkitehti esimerkiksi päivittää säännöllisin väliajoin arkkitehtimallin IFC-tiedostona projektipankkiin, mistä rakennesuunnittelija pääsee lataamaan sen oman rakennemallinsa referenssiksi. Projektipankkiin tallennetaan sekä virallisia 2D-dokumentteja että 3D-malleja. Hankkeen alkuvaiheessa epävirallisia suunnitelmia jaetaan yhä myös sähköpostin välityksellä. (Pro IT 2004, s. 35)

Alalla vallitsevat tietomallinnuksen yleiset ohjeet ja säännöt on koottu vuonna 2007 julkaistuun ja vuosina 2011–2012 päivitettyyn Yleisten tietomallivaatimukset 2012 -ohjeeseen. Ohje jakautuu 14 osaan, joissa esitellään tietomallinnuksen hyödyntämismahdollisuuksia sekä eri osapuolien mallinnusvaatimuksia. Julkaisun kolmannessa osassa käydään läpi arkkitehtisuunnittelun yleisiä mallinnusvaatimuksia, joista selviää hyvin arkkitehtisuunnittelun vaiheet ja niiden sisältö. Hankkeen alussa on tärkeä sopia tarkasti yhteiset säännöt mallinnuksen ja suunnittelun toteutukseen. Kun arkkitehti, rakennesuunnittelija sekä muut suunnitteluosapuolet ovat tietoisia yhteisistä toimintatavoista voidaan välttyä monien ristiriitaisuuksien ja muiden ongelmien muodostumiselta. (RT 10-11080 2012)

Ennen mallinnuksen aloitusta arkkitehti sopii yhdessä muiden osapuolien kanssa projektissa käytettävästä koordinaatistosta, jotta tietomallien yhdistäminen onnistuu ongelmitta.

Arkkitehti jakaa mallin usein myös lohkoihin ja kerroksiin suunnittelutiedon hallinnoinnin helpottamiseksi. Arkkitehti mallintaa tyypillisesti kaikki maanpäälliset osat, mutta esimerkiksi perustukset jäävät rakennesuunnittelijan vastuulle. Mallintamisen tarkkuustaso riippuu aina hankkeen vaiheesta ja projektin alussa yhteisesti sovituista mallinnusperiaatteista. Tarkkuusvaatimukset voidaan jakaa kolmeen tasoon, jotka ovat:

Taso 1 Rakennusosat on mallinnettu sijainnin ja geometrian osalta oikein sekä nimetty kuvaavasti. Mallia käytetään pääasiassa suunnitelmien yhteensovittamiseen ja suunnittelijoiden väliseen vuorovaikutukseen.

Taso 2 Rakennusosien sijainti- geometria ja määrätiedot ovat oikein ja rakennetyypit on määritetty ja nimetty. Mallia käytetään hanke- ja luonnosvaiheen analyyseihin sekä tuoteosien määrälaskentaan.

Taso 3 Rakennusosien sijainti, geometria ja tarvittavat tuotetiedot on määritetty tarkasti. Mallia käytetään työmaan aikataulutukseen ja hankintoihin.

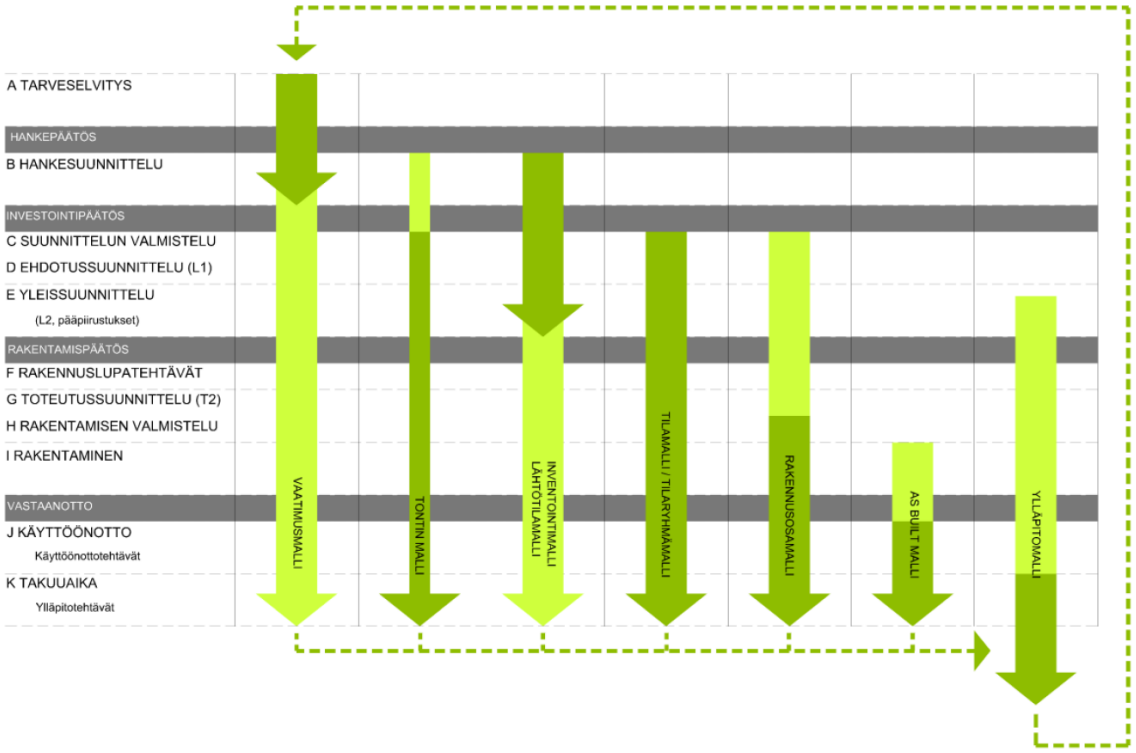
Arkkitehtimallin tietosisältö tarkentuu suunnitteluprosessin edetessä. Rakennusosille voidaan määrittää erilaisia tarkkuusvaatimuksia, minkä seurauksena eri rakennusosien mallinnustarkkuudessa saattaa ilmetä eroja. (RT 10-11068 2012)

Rakennesuunnittelija saa arkkitehdilta tietomallin, josta selviää ennen kaikkea tilojen ja rakenteiden tilavaraukset. Arkkitehti ja rakennesuunnittelija sopivat yhteistyössä käytettävistä rakennetyypeistä, minkä jälkeen arkkitehti mallintaa tilat sovittujen rakennetyyppien mukaan. Rakennesuunnittelija vastaa erityisesti kantavien rakenteiden suunnittelusta, joten arkkitehti mallintaa ne ensisijaisesti vain näkyvinä pintoina eli niin sanottuina tilavarauskappaleina. Lisäksi arkkitehti mallintaa alustavat aukotukset nimellismitoilla. Rakennesuunnittelijan suunnittelemien rakenteiden tulisi tämän jälkeen sopia niille suunniteltujen tilavarausten sisään. Kun rakennesuunnittelijan suunnitelmat tarkentuvat, arkkitehti päivittää muuttuneet rakenteet tarvittavalla tarkkuudella myös arkkitehtimalliin. (RT 10-11068 2012)

Rakennesuunnittelijan kannalta merkittävimmät mallinnusvaiheet kuuluvat ehdotus-, yleis- ja toteutussuunnitteluun. Ehdotussuunnitteluvaiheessa arkkitehti luo kohteesta tilamallin, joka sisältää tavallisesti tilat, niitä rajaavat seinät, lattiat ja katot sekä kunkin tilan käyttötarkoituksen. Tavoitteena on tehdä mallista IFC-tiedosto, josta selviävät luokitellut ja numeroidut tilat sekä pinta-ala- ja tilavuustiedot. Yleissuunnittelussa mallinnuksessa käytetään yleensä tarkkuustasoa 1. Arkkitehti luo tilamallin pohjalta alustavan rakennusosamallin, johon luodaan sijainnin ja geometrian kannalta mittatarkat rakennusosat. Rakennusosille annetaan myös karkeat nimikkeistön mukaiset tyyppimerkinnät, kuten esimerkiksi US, VS, AP tai YP. (RT 10-11068 2012)

Lopulta toteutussuunnittelussa rakennusosamalli päivitetään toteutettaviksi rakenteiksi ja tuoteosiksi. Tarkkuustaso on tavallisesti 1 tai 2, mutta myös tasoa 3 voidaan joissakin tapauksissa käyttää. Toteutussuunnittelussa kaikki rakennusosat, kuten seinät, laatat, palkit, pilarit, portaat, ovet ja ikkunat tulee mallintaa niille tarkoitetuilla työkaluilla. Rakennusosat eivät myöskään saa törmätä toisiinsa eikä niiden väliin saa jäädä turhia rakoja.

Pystyrakenteet kuten seinät mallinnetaan osissa kerroskorkeuden mukaisesti ja malliin tehdään varaukset esimerkiksi alaslasketuille katoille. Arkkitehti päivittää rakennusosiin myös rakennusselostuksen mukaiset tyyppitiedot sekä pintamateriaalit. Kaikki rakennushankkeessa käytettävät tietomallit ja niiden vaiheistus on esitetty kuvassa 6. (RT 10-11068 2012)



Kuva 6. *Rakennushankkeen tietomallirakenne (RT 10-11068 2012, s. 6)*

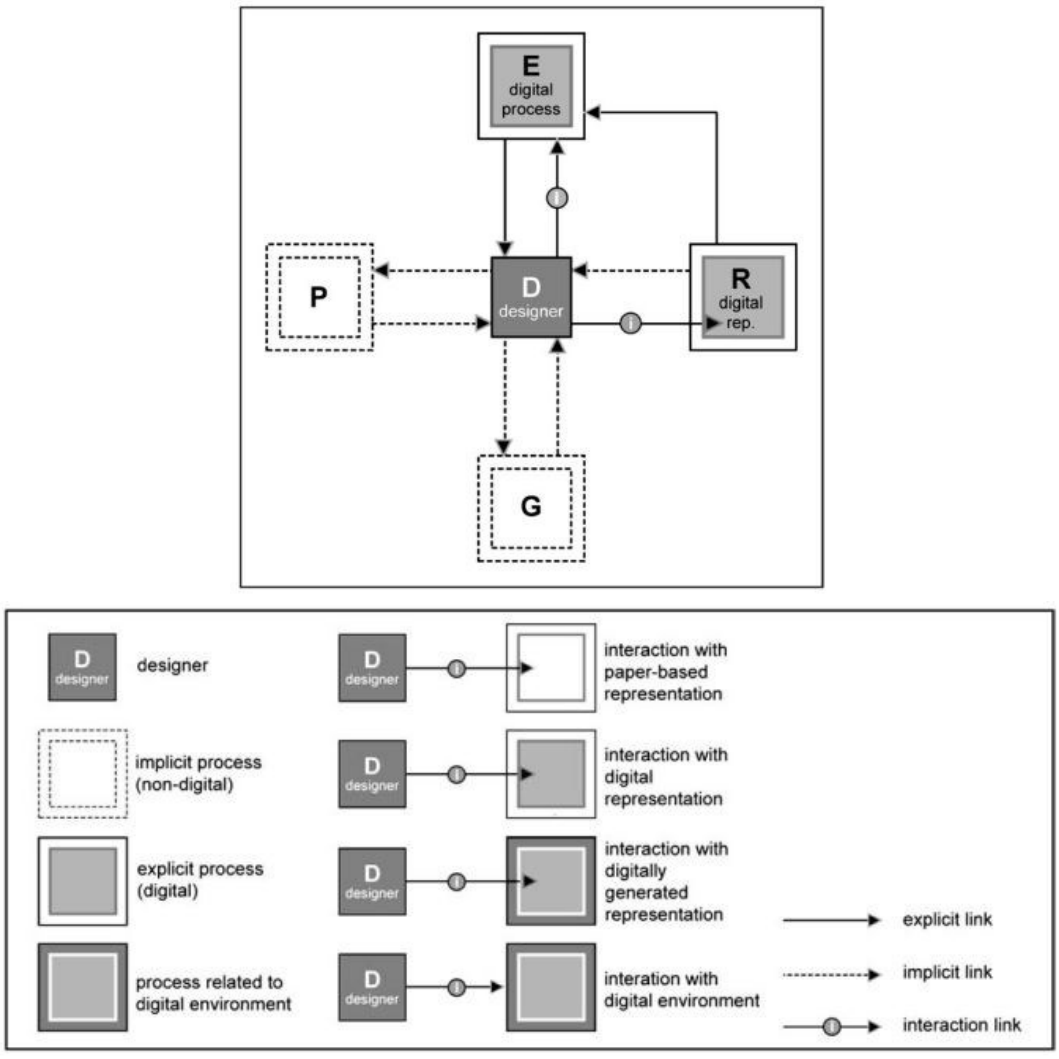
Arkkitehdin ja rakennesuunnittelijan tulee käydä jatkuvaa vuoropuhelua suunnitteluprosessin edetessä. Suunnitelmien muuttuessa on toisen osapuolen saatava siitä tieto, jotta hän voi päivittää omat suunnitelmansa vastaamaan muutoksia. Näin voidaan välttyä ristiriitaisilta tietomalleilta ja piirustuksilta.

2.3 Rakenteiden mitoitus, mallinnus ja raportointi

Suunnittelussa käytetään nykyään paljon hyödyksi erilaisia tietoteknisiä sovelluksia ja siten myös koko suunnitteluprosessia voidaan pitää lähes täysin digitaalisena. Rakennusten digitaalinen suunnitteluprosessi voidaan luokitella Oxmanin (2006) mukaan suunnittelijan, suunnittelutavoitteen ja käytettävän suunnitteluprosessin ominaisuuksien perusteella viiteen erilaiseen suunnittelumalliin:

1. CAD-mallit (engl. CAD models).
2. Muodostumismallit (engl. Formation models).
3. Generatiiviset mallit (engl. Generative models).
4. Suorituskykymallit (engl. Performance models).
5. Integroidut yhdistelmämallit (engl. Integrated compound models).

Oxman kuvaa erilaisia digitaalisia suunnitteluprosesseja symbolisten mallien avulla. Visualisoinneissa esitetään suunnittelija sekä suunnittelun neljä komponenttia: esittely (representation, R), generointi (generation, G), arviointi (evaluation, E) ja suorituskyky (performance, P). Lisäksi kaavioissa kuvataan suunnittelijan ja edellä esitettyjen komponenttien välisiä suhteita ja niiden ominaisuuksia. Nykyinen rakennesuunnitteluprosessi noudattaa pääasiassa Oxmanin luettelon ensimmäistä kohtaa, CAD-mallia. Kuten kuvasta 7 nähdään, suunnittelija muodostaa suunniteltavasta kohteesta tarkan digitaalisen tietomallin ja sitä voidaan hyödyntää myös suunnittelun arviointiin ja erilaisten analyysien tekemiseen. Suunnittelun generointi eli esimerkiksi tietomallin luominen ja muokkaaminen sekä rakenteiden suorituskyvyn varmistus toimivat kuitenkin yhä manuaalisesti. (Oxman 2006, s. 246–249)



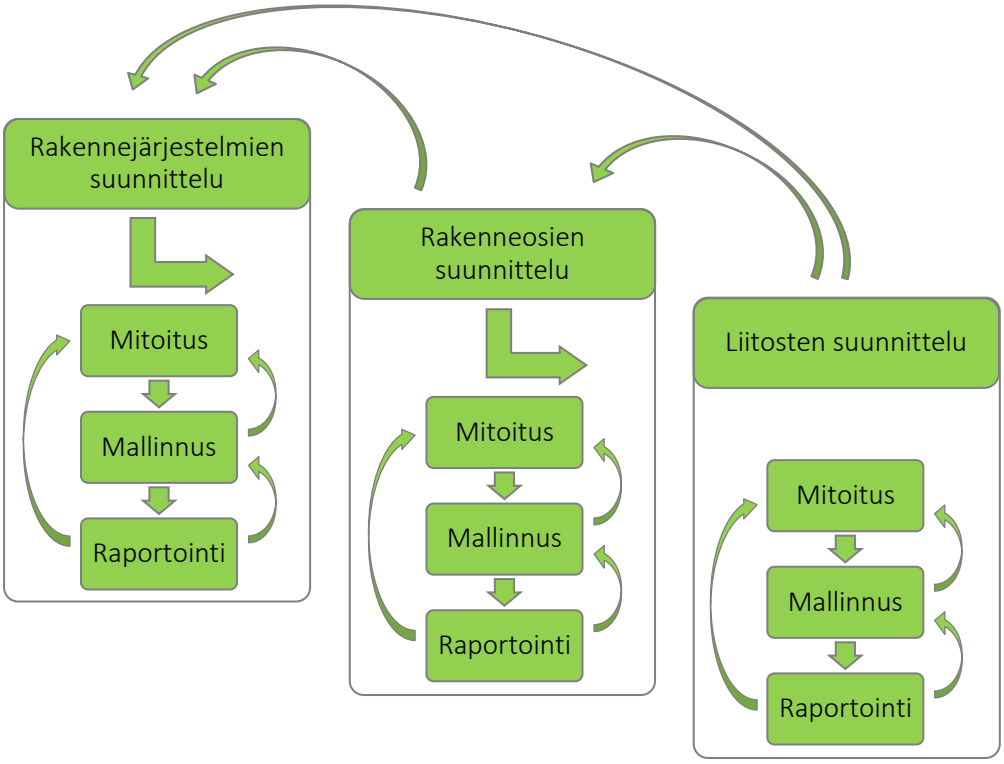
Kuva 7. *Perinteisen rakennesuunnitteluprosessin ominaisuudet (Oxman 2006).*

Rakennesuunnittelun sisällön voi karkeasti jakaa kolmeen osaan: mitoitukseen, mallinnukseen ja raportointiin. Oxmanin prosessikaaviossa esitettävistä komponenteista voi myös löytää nämä tekijät. Suorituskykyä voidaan pitää mitoituksena, arviointia mallin-

nuksena ja esittelyä raportointina. Lisäksi generoinnilla kuvataan näiden suunnittelutehtävien toteutustapaa. Nämä työvaiheet voivat tapahtua osittain muussakin järjestyksessä, mutta pääasiassa ensin tapahtuu mitoitus, sitten mallinnus ja lopulta raportointi. Tietomallinnuksen tai lyhyemmin mallinnuksen voi katsoa kuuluvan myös osaksi raportointia, mutta nykyään sen ollessa merkittävä osa suunnittelua tarkastellaan sitä tässä työssä omana kokonaisuutenaan. Joissain kohteissa, joissa tietomallinnusta ei käytetä, jää mallinnusvaihe kokonaan pois. Yksinkertaistetusti rakennesuunnittelu on siis rakennejärjestelmien, rakenneosien ja liitosten suunnittelua ja suunnittelun raportointia tietomallien, piirustusten ja muiden suunnitteludokumenttien avulla.

Nykyään monet rakennesuunnitteluun kuuluvat tehtävät voidaan suorittaa erilaisia tietokoneohjelmia hyväksi käyttäen. Ohjelmistojen käyttö helpottaa ja nopeuttaa suunnittelua, mutta toisaalta se myös saattaa luoda paljon sellaisia työtehtäviä, joita ei ole ennen ollut. Esimerkiksi tietomallinnus vie suuren osan suunnittelijoiden työajasta, mutta toisaalta se myös nopeuttaa esimerkiksi piirustusten tekoa huomattavasti. Vaikka suunnittelijoilla on käytössään monia erilaisia työtä helpottavia ohjelmia, on suunnittelijan ammatillinen pätevyys edelleen erittäin tärkeää. Ohjelmat toimivatkin ainoastaan työkaluina, joiden avulla rakennesuunnittelija tekee työtään. Ohjelmistot keskittyvät usein vain johonkin tiettyyn rajattuun aiheeseen, minkä vuoksi suunnittelijan on myös huolehdittava eri osakokonaisuuksien yhteensovittamisesta. Yleisesti ottaen mitä laajempi kokonaisuus on kyseessä, sitä vähemmän suunnitteluohjelmia pystytään hyödyntämään. Esimerkiksi tietyn yksittäisen rakenneosan mitoitus onnistuu ohjelmistojen avulla melko helposti, mutta koko rakennuksen toteutettavuuden arvioinnissa suunnittelijalla ei ole tietoteknisiä työkaluja apunaan.

Nykyään suunnitteluohjelmistojen kehittyessä ja monipuolistuessa suunnittelun sisältö ei ole enää yhtä selkeä kuin ennen. Rakenteiden mitoituksessa saatetaan esimerkiksi hyödyntää FEM-ohjelmaa, jossa rakennejärjestelmä ensin mallinnetaan ja vasta tämän jälkeen osat mitoitetaan. Ohjelmistojen avulla myös suunnitelmien muuttaminen on helpottunut huomattavasti. Tämän seurauksena rakennesuunnittelu ei noudata aina järjestystä mitoitus, mallinnus ja raportointi vaan suunnittelun sisältö vaihtuu useita kertoja suunnittelukokonaisuuden sisällä. Rakennesuunnittelu onkin iteratiivinen prosessi niin yksittäisten suunnitteluosien kuin suunnittelukokonaisuudenkin kannalta. Tämän vuoksi on tärkeää, että suunnitelmien muutostenhallinta on mahdollisimman helppoa ja tehokasta. Rakennesuunnitteluprosessin vaiheita ja sisältöä on tarkennettu kuvassa 8.



Kuva 8. *Rakennesuunnitteluprosessin sisältö ja vaiheet.*

2.3.1 Rakenteiden mitoitus

Ennen kuin mitään rakenteita voi alkaa mitoittaa, pitää suunnittelijoiden päättää, millaisista rakenneosista arkkitehdin suunnittelemat tilat muodostetaan. Käytettävien rakenneosien materiaalit ja perustyypit valitaan tavallisesti yhteistyössä tilaajan, arkkitehdin ja rakennesuunnittelijan kesken, jotta saadaan luotua taloudellisia, turvallisia ja esteettisiä ratkaisuja. Kun rakenteiden perusperiaatteet on päätetty, voidaan rakenteiden mitoitus aloittaa. Rakenteiden mitoitustapa vaihtelee aina tarkasteltavan rakenneosan ominaisuuksien mukaan, mutta perusperiaatteet ovat kuitenkin samat.

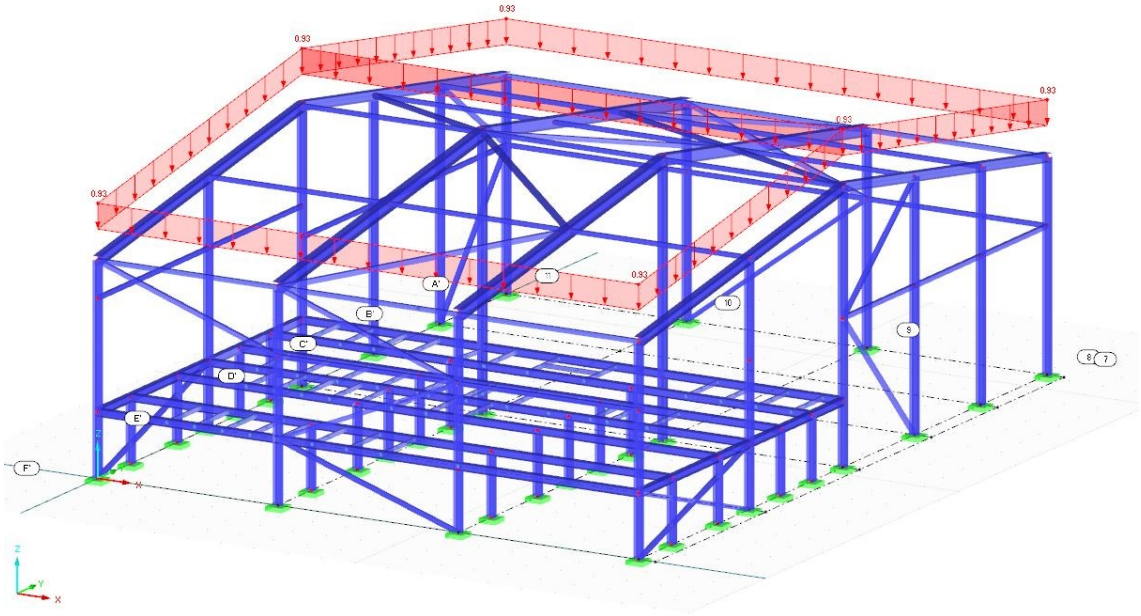
Rakenteiden mitoitus on suunnitteluprosessin tarkimmin säädelty tehtävä, koska sen suoritus vaikuttaa merkittävimmin rakennuksen turvallisuuteen. Suunnittelua ohjataan maankäyttö- ja rakennuslaissa (MRL 120§) sekä ympäristöministeriön asetuksessa kantavista rakenteista (477/2014). Käytännössä kaikki lakien ja määräysten vaatimukset täyttyvät, kun rakennelaskelmat tehdään eurooppalaisten suunnittelustandardien eli eurokoodien mukaan. Lisäksi suunnittelussa tulee huomioida eurokoodien kansalliset liitteet sekä käytettävien rakennustuotteiden tuotestandardit. (RIL 229-1-2013)

Kun rakenteiden perusperiaatteet on päätetty, aloitetaan rakennelaskelmien teko. Ensin määritetään rakenteisiin kohdistuvat kuormitukset. Suomessa rakenteisiin tyypillisesti kohdistuvia kuormituksia ovat rakenteiden oma paino, hyötykuorma, lumikuorma ja tuu-

likuorma. Lisäksi kuormitusta aiheuttavat muun muassa maan- ja pohjavedenpaine, lämpötilan muutokset sekä nosturit ja muut koneet. Kuormitusten määrityksessä tulee ottaa huomioon myös erilaiset tilanteet, joissa kuormituksia syntyy. Rakenteiden kestävyys tuleekin tarkastella toteutus-, käyttö-, palo- ja onnettomuustilanteessa. (RIL 229-1-2013, s. 68–69)

Kun kuormitukset on määritetty, luodaan tarkasteltavasta kokonaisuudesta laskentamalli. Rakennesuunnittelija tutkii kyseiseen kokonaisuuteen kohdistuvat kuormat ja syöttää ne käytettävään laskentamalliin. Nykyään rakenteiden mitoituksessa käytetään paljon hyödyksi erilaisia tietokoneohjelmia, kuten esimerkiksi FEM-, statiikka- ja mitoitusohjelmia. Laskelmia tehdään edelleen myös käsin, mutta pääasiassa vain laskentaohjelmien tulosten tarkistamista varten. Laskentamallin ominaisuudet vaihtelevat suunniteltavan kokonaisuuden mukaan, mutta pääpiirteet ovat aina samat. Tyypillisesti mitoitus etenee laajemmista rakennejärjestelmistä kohti tarkkoja detaljeja ja tämän seurauksena myös käytettävien laskentamallienmallien koko pienenee usein suunnittelun edetessä. (RIL 229-1-2013)

Laskelmien teko aloitetaan tyypillisesti rakennuksen stabiliteetin tutkimisella. Siinä tarkastellaan erityisesti rakennuksen jäykistystapaa ja jäykistävien rakenteiden kestävyyttä. Rakennuksen rungosta luodaan tyypillisesti 3D-malli, jonka avulla voidaan varmistaa rungon riittävä jäykistys. Laskentamallin luonnissa on tärkeää määrittää kuormien todenmukainen jakautuminen rakenneosille, rakenteiden oikeat laskennalliset mitat ja tuennat sekä rakenneosien ja niiden välisten liitosten oikeat jäykkyysominaisuudet. Laskentamallin ominaisuudet pyritään määrittämään siten, että mallin toiminta vastaa mahdollisimman hyvin todellisen rakenteen toimintaa. Alustavan laskentamallin avulla voidaan päättää käytettävä jäykistystapa ja rakenneosien ominaisuudet karkealla tasolla, vaikka rakenneosia ei vielä mitoitettaisi. Esimerkki rakennuksen laskentamallista on esitetty kuvassa 9. (RIL 229-1-2013, s. 72)



Kuva 9. *Rakennejärjestelmän laskentamalli.*

Rakennuksen stabiliteetin varmistuttua siirrytään mitoittamaan yksittäisiä rakenneosia. Rakenneosien mitoituksessa voidaan hyödyntää stabiliteetin tutkimiseen käytettyä mallia tai rakenneosasta voidaan tehdä myös oma laskentamalli. Rakenneosan laskentamallin teossa tulee huomioida samat tekijät kuin suurempien rakennejärjestelmien malleissa. Erityisen tärkeää on määrittää kaikki osaan vaikuttavat kuormat oikein. Monimutkaisten rakennejärjestelmien kohdalla osaan vaikuttavien kuormitusten määrittäminen saattaa olla melko haastavaa. Tämän vuoksi rakennejärjestelmän laskentamallia kannattaa usein hyödyntää esimerkiksi rakenneosien voimasuureiden määrittämiseen. (RIL 229-1-2013)

Erityisesti tietokoneohjelmia käytettäessä on tärkeää muistaa määrittää kaikki rakenneosan ominaisuudet, kuten esimerkiksi tukien määrä ja laatu sekä puristettujen rakenteiden nurjahduspituus. Kun rakenneosan tuenta, kuormitukset ja niiden aiheuttamat voimasuureet on saatu määritettyä, suoritetaan itse mitoitus. Mitoitus vaihtelee hyvin paljon rakenneosan materiaalin ja rakennetyypin mukaan, mutta käytännössä mitoituksessa varmistetaan, että rakenneosan jännitykset eivät ylitä osan lujuutta. Mitoituksen seurauksena saadaan rakenneosan materiaali- ja profiiliominaisuudet, kuten esimerkiksi poikkileikkauksen koko, raudoitteiden määrä ja betonilaatu. Rakenneosat mitoitetaan tyypillisesti murto- ja käyttörajatilassa. Lisäksi tulee varmistaa rakenneosan kestävyys asennusvaiheessa ja palotilanteessa. (RIL 229-1-2013, s. 68–72)

Kun rakenneosat on saatu mitoitettua, lasketaan vielä osien välisten liitosten kestävyys. Liitosten mitoitus vaihtelee hyvin paljon liitostyypin mukaan. Koska liitosten todellisen toiminnan analysoiminen on haastavaa, käytetään mitoituksessa usein yksinkertaistettuja varmalla puolella olevia menetelmiä. Suunnittelijat pyrkivät käyttämään liitoksissa tyyp-

piratkaisuja, joiden toimivuus tunnetaan ja joiden toteutus on tavallisesti kustannustehokasta. Tavallisista liitostyypeistä löytyy usein myös suunnittelua helpottavia mitoitustaulukkoja, mikä helpottaa suunnittelijan työtä.

Kuten kuvassa 8 esitetään, joudutaan suunnitteluprosessin vaiheissa palamaan usein myös taaksepäin. Esimerkiksi kun rakenneosat on saatu mitoitettua, tulee suunnittelijan varmistaa, että rakennejärjestelmän jäykistys on edelleen kunnossa. Kun rakenneosien profiileja on muutettu, saattaa kuorma siirtyä rakenteissa eri tavalla ja rakennejärjestelmä ei enää toimi kokonaisuutena. Rakenteiden mitoitus on siis iteratiivinen prosessi, jossa pyritään valitsemaan mahdollisimman tehokkaasti toimivat rakenneosat ja liitokset siten, että rakennekokonaisuus toimii. Rakennesuunnittelija valitsee käytettävien rakenteiden profiilit ja muut ominaisuudet pääasiassa kokeilemalla, joten suunnittelun tehokkuus on vahvasti riippuvainen suunnittelijan asiantuntemuksesta. Myös erilaisten laskentaohjelmien käyttö nopeuttaa nykyään eri rakennevaihtoehtojen kokeilua, ja tekee rakenteiden mitoituksesta näin nopeampaa.

2.3.2 Rakenteiden tietomallinnus

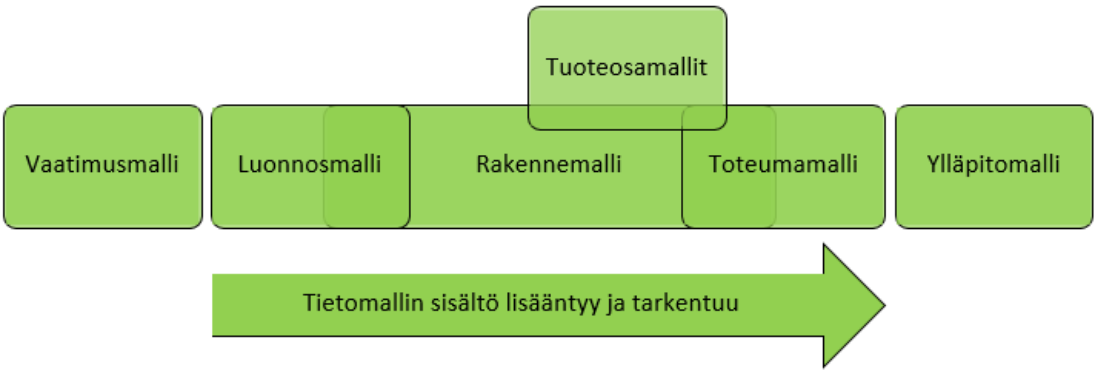
Vaikka Building Information Modelling (BIM) eli tietomallinnus on rakennusalalla vielä suhteellisen uusi asia, luotiin sen perusteet jo 1980-luvulla. Ennen tietomallinnusta suunnittelu tapahtui pääasiassa useiden itsenäisten kaksiulotteisten suunnitelmien avulla, mikä aiheutti usein ristiriitaisuuksia suunnitelmien välille. Tietomallinnus kehitettiin vastaamaan tähän ongelmaan yhtenäisen kolmiulotteisen mallin avulla. Siinä yhdistyvät rakenteiden kolmiulotteinen geometria sekä rakenteisiin liitettävät ominaisuustiedot. Kun rakenteita alettiin tietomallintaa, erillisten suunnitelmien väliset ristiriitaisuudet vähenivät ja suunnittelukokonaisuuksien hallitseminen helpottui. (Aish 2013, s. 40–43)

Rakennesuunnittelijat tuottavat nykyään paljon erilaisia tietomalleja, joita kutsutaan yleisesti myös rakennemalleiksi. Rakennesuunnittelijan luomien mallien tarkkuus ja muut sisältövaatimukset vaihtelevat aina suunnittelukohteesta ja -vaiheesta riippuen, mutta yleisiä määrittelyjä voidaan silti tehdä. Rakennemallissa esitetään tyypillisesti vähintään kantavat rakenteet, ei-kantavat betonirakenteet sekä koon ja sijainnin kannalta merkittävät rakennusosat. Rakenneosat tulee mallintaa tarkoituksenmukaisilla työkaluilla, jotta osien ominaisuudet välittyvät oikein kaikille mallia käyttäville osapuolille. Rakenneosille syötetään usein myös muita ennalta yhdessä määriteltyjä tietoja, kuten osien lohko-, kerros-, numerointi-, nimi- ja valmiusastetietoja. Kaikki rakenneosiin liitettävä data helpottaa tietomallin käsittelyä ja lisää sen hyödyntämismahdollisuuksia, minkä johdosta tietomallit ovat muuttuneet tietosisällöltään hyvin vaativiksi kokonaisuuksiksi. (RT 10-11070 2012, s. 2–3)

Rakennesuunnittelun tietomallinnus aloitetaan yleensä aina uudesta tyhjästä tiedostosta, vaikka samankaltaisia rakenteita olisi mallinnettu aikaisemmissa suunnittelukohteissa. Vanhojen tietomallien hyödyntäminen on hyvin vaikeasti hallittavaa ja työlästä, minkä

vuoksi on usein tehokkainta aloittaa tietomallin rakentaminen tyhjään tiedostoon. Rakennesuunnittelija saa tietomallinsa lähtötiedoksi vaatimusmallin, joka voi olla tietomalli, piirustus, tekstiasiakirja, taulukkomuotoinen esitys tai näiden yhdistelmä (RT 10-11070 2012, s. 3). Ensin rakennesuunnittelijan tietomalliin luodaan hankkeen muiden osapuolien kanssa yhteisesti sovittu moduuliverkko. Moduuliverkko toimii suunniteltavien rakenteiden sijaintitiedon sääntönä, jota kaikkien rakennushankkeeseen osallistuvien tulee ehdottomasti noudattaa. Kun moduuliverkko ja siihen sidotut laitteet ja rakenteet sijaitsevat kaikilla hankeen osapuolilla samassa koordinaatistossa, on eri tietomallien yhdistäminen ja hyödyntäminen mahdollista.

Kun tietomallin sijainti on saatu määritettyä tarkasti, voidaan aloittaa rakenneosien mallinnus. Rakenteiden mallinnus voidaan jakaa erinimisiin vaiheisiin, vaikka todellisuudessa eri mallinnusvaiheiden välissä ei olekaan selvää rajapintaa. Tietomallin rakenneosat tarkentuvat suunnittelun edetessä eikä mallin tarkkuustasoa ja vaihetta voi yksiselitteisesti määritellä. Yleisesti ottaen tietomallin vaiheina voi kuitenkin pitää luonnos-, rakenne ja toteumamallia sekä erikseen tuotettavia tuoteosamalleja. Lisäksi mallinnuksen lähtötietona on vaatimusmalli ja lopputuloksena ylläpitomalli. Rakennesuunnittelijan tietomallivaiheet on esitetty kuvassa 10.

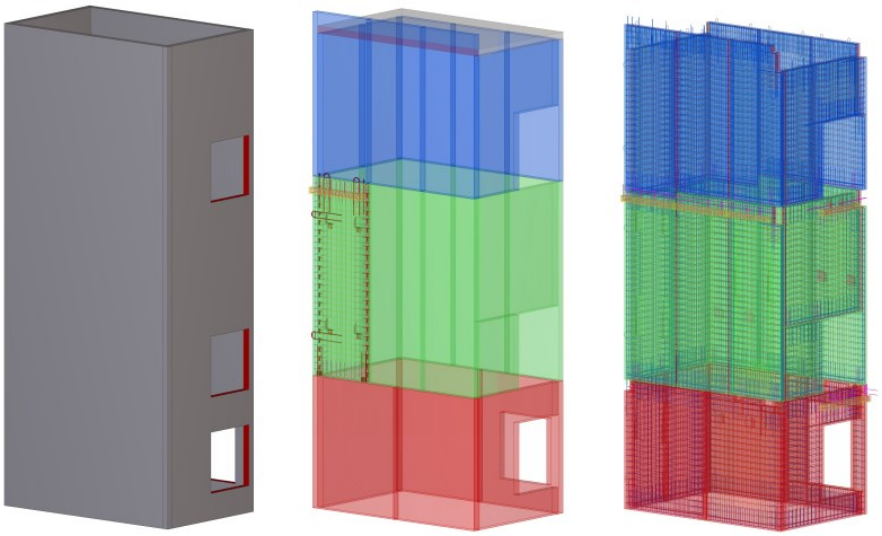


Kuva 10. *Rakennesuunnittelun tietomallinnuksen vaiheet perustuen lähteeseen (Pro IT 2004).*

Rakennesuunnittelija saa yleensä arkkitehtimallin oman suunnittelunsa referenssiksi, ja voi sen pohjalta alkaa luoda alustavaa rakennemallia eli luonnosmallia. Luonnosmalliin määritetään perustukset ja runkorakenteet perusgeometrian ja sijainnin osalta oikein, mutta esimerkiksi raudoitteita tai liitoksia ei vielä mallinneta. Tässä vaiheessa tietomalliin voidaan integroida myös laskentamalli, jonka avulla varmistetaan käytettävien rakenneosien kestävyys. (RT 10-11070 2012)

Kun luonnosvaihe on ohi ja suunnitelmat ovat tarkentuneet, aletaan luonnosmallista muodostaa varsinaista rakennemallia. Aluksi muodostetaan hankintoja palveleva rakennemalli, jossa eri rakennusosatyypeistä tehdään mallielementit ja -kokoonpanot. Esimerkiksi betonipilarit luokitellaan pilarityypeittäin ja kustakin tyypistä mallinnetaan yksi

mallielementti, joka on geometrian, sijainnin, liittymien, raudoitteiden ja valutarvikkeiden osalta oikein. Kaikista mallielementistä tuotetaan myös tyyppipiirustukset hankintoja varten. Kun mallielementit on saatu valmiiksi, jatkaa rakennesuunnittelija muiden osien mallintamista samaan tarkkuuteen. Tässä toteutusta palvelevassa suunnitteluvaiheessa rakennemalli muokataan niin tarkaksi, että sen avulla voidaan tilata tarvittavat osat ja toteuttaa suunnitellut rakenteet työmaalla. Jos jokin osakokonaisuus suunnitellaan toisen osapuolen toimesta, tulee suunnittelijoiden sopia yhteistyöstä ja tarvittavien tietomallien jakamisesta ja yhteensovittamisesta. Luonnosmallin sekä hankintoja ja toteutusta palvelevien rakennemallien tietosisältöä on kuvattu kuvassa 11. (RT 10-11070 2012)



Kuva 11. *Luonnosmallin sekä hankintoja ja toteutusta palvelevien rakennemallien tietosisältö (RT 10-11070 2012).*

Toteutussuunnittelun jälkeen rakennemalli on valmis. Jos rakentamisen aikana rakenteisiin tehdään vielä merkittäviä muutoksia, tulee ne päivittää toteumamalliin. Toteumamalliin voidaan päivittää myös esimerkiksi ulkopuolisten suunnittelijoiden tekemät erilliset tuotemallit. Yleensä rakennemalli on kuitenkin ajan tasalla hankkeen päättyessä, jolloin erilliselle toteumamallille ei ole tarvetta. Rakennesuunnittelijan tietomallien ja toteutuneiden rakenteiden yhtäläisyys tulee varmistaa, jotta tietomallia voidaan hyödyntää luotettavasti myös rakennuksen elinkaaren aikana. (RT 10-11070 2012, s. 9)

Rakennesuunnittelijan tietomallinnuksessa tulee muistaa huomioida myös muiden osapuolien muuttuvat suunnitelmat. Rakennesuunnittelijan tulee varmistaa, että rakennemalli vastaa aina ajantasaista vaatimusmallia. Pääasiassa vaatimusmallina toimivat arkkitehtisuunnitelmat, mutta myös tilaajan ja muiden hankeosapuolten muuttuvat tarpeet tulee huomioida. Rakennesuunnittelijan tulee myös varmistaa, että käytetyt laskentamallit vastaavat tietomalliin määritettyjä rakenteita. Lisäksi rakennesuunnittelijan on hyvä päivittää ja jakaa oma tietomallinsa muille hankkeen osapuolille riittävän usein. Näin

kaikki osapuolet pysyvät ajan tasalla toistensa suunnitelmista ja voivat tarkentaa omia suunnitelmiaan kohti lopullisia ratkaisuja.

2.3.3 Rakennesuunnittelun raportointi

Rakennesuunnittelun raportointia voidaan tehdä esimerkiksi kirjallisten asiakirjojen, piirustusten, luetteloiden ja laskelmien avulla. Nykyään lähes kaikki suunnitteluasiakirjat tuotetaan erilaisia tietokoneohjelmia, kuten mallinnus-, taulukkolaskenta- ja mitoitusohjelmia, hyödyntäen. Kun suunnittelu muuttuu yhä enemmän tietomallipohjaiseksi, voidaan suunniteltavia rakenteita havainnollistaa myös kolmiulotteisen mallin avulla. Lisäksi tietomallista voidaan tehokkaasti tuottaa monia tarvittavia asiakirjoja sekä myös täysin uudentyyppisiä raportteja. Suunnitteluasiakirjoja luotaessa tulee huomioida, että tuotettavien asiakirjojen tiedot esitetään yksiselitteisesti eivätkä ne sisällä turhaa tietoa tai ristiriitaisuuksia. Asiakirjojen tulee lisäksi olla selkeitä ja käyttötarkoitukseen sopivia, jotta niiden tietosisältöä voidaan hyödyntää ongelmitta. (RIL 229-1-2013, s. 23–24)

Suunnittelun raportointi on tärkeä osa suunnittelua, koska suunniteltavista rakenteista tulee saada tieto myös hankkeen muille osapuolille esimerkiksi hankintaa ja valmistusta varten. Vaikka suunnitteluasiakirjoja tuotetaan pääasiassa valmiiksi suunnitelluista rakenteista, voidaan erilaisia raportteja ja analyysejä tehdä jo suunnittelun aikana. Erityisesti tietomallit mahdollistavat suunniteltavien rakenteiden analysoinnin heti suunnittelun alusta aina käyttöön ja ylläpitoon. Tietomalleja voidaan hyödyntää esimerkiksi määrä- ja kustannustiedon laskentaan, energia- ja ympäristöanalyyseihin, suunnittelun havainnollistamiseen ja rakennettavuuden tarkasteluun sekä laadunvarmistukseen ja törmäystarkasteluihin. Tietomallit mahdollistavat myös uudenlaisen tavan raportoida suunnitelluista rakenteista tietomallipohjaisesti. Esimerkiksi tiedot suunnitelluista ontelolaatoista voidaan nykyään lähettää tehtaalle IFC-muodossa, jolloin erillisiä lappukuvia ei tarvita. (RIL 229-1-2013, s. 38–39)

2.4 Suunnittelun kustannusvaikutukset

Rakennusten suunnittelukustannukset ovat suhteellisen pienet verrattuna rakennushankkeen kokonaiskustannuksiin. Rakennuksen koosta ja käyttötarkoituksesta riippuen, suunnittelun kustannukset voivat olla noin 4–10 % hankkeen kokonaiskustannuksista. Kun suunnittelukustannukset jaetaan vielä toimialoittain, on rakennesuunnittelun osuus noin 26 % suunnittelukustannuksista eli vain 1,0–2,6 % kokonaiskustannuksista. (Haahtela & Kiiras 2014, s. 311) Toisaalta vaikka suunnittelukustannuksien osuus kokonaiskustannuksista on suhteellisen pieni, määräytyvät hankkeen kustannukset lähes kokonaan suunnitteluvaiheessa. Erityisesti suunnittelun alkuvaiheessa tehtävien ratkaisujen merkitys korostuu hankkeen taloudellisuuden hallinnassa. Tämän vuoksi onkin suositeltavaa, että hankkeen eri osapuolet osallistuvat suunnitteluun jo varhaisessa vaiheessa. Näin myös alustavasta kustannusarviosta saadaan tarkempi ja luotettavampi.

Rakennushankkeen kustannushallinta voidaan karkeasti jakaa kolmeen päävaiheeseen: ohjelma-, suunnittelu- ja toteutusvaiheeseen. Hankkeen sisältöön ja kustannuksiin vaikuttamisen mahdollisuudet vähenevät selvästi, kun edetään ohjelmavaiheesta, suunnitteluun ja siitä edelleen toteutukseen. Eniten kustannuksiin voi vaikuttaa ohjelmavaiheessa, jossa päätetään hankeen laajuus- ja laatutavoitteet ja siten myös aikataulu- ja kustannustavoitteet. Tämän jälkeen suunnitteluvaiheessa pyritään löytämään ratkaisut, joiden avulla asetetut tavoitteet voidaan saavuttaa. Suunnitteluvaiheessa voidaan vielä tarkastella erilaisia ratkaisuvaihtoehtoja ja vertailla niiden ominaisuuksia ja vaikuttaa näin merkittävästi kustannuksiin. Toteutusvaiheessa rakenteet on suunniteltu ja kustannuksiin voidaan vaikuttaa enää vain valittavilla tuotantoratkaisuilla, kuten esimerkiksi työmenetelmillä ja toteutusaikataululla. Toteutusvaiheessa tapahtuvilla päätöksillä ei voida kuitenkaan merkittävästi enää vaikuttaa kokonaiskustannuksiin. (Lindholm 2009, s. 9–11)

Hankkeelle on aina asetettava tarkat tavoitteet, jotta pystytään suunnittelemaan mahdollisimman hyvin tilaajan ja käyttäjän tarpeita vastaama kokonaisuus. Tavoitteet voivat olla mitattavia, kuten tarvittavien tilojen määrä ja laatu, aiheutuvat kustannukset ja hankkeen aikataulu tai ei-mitattavia, kuten tilojen esteettisyys, dynaamisuus ja muuntojoustavuus. Tilaajan tulisi saada erityisesti hankkeen alkuvaiheessa riittävästi tietoa erilaisten vaihtoehtojen ominaisuuksista ja kustannuksista, jotta hän tietää tekemiensä päätösten todelliset vaikutukset. Tämä tarkoittaa sitä, että suunnittelijoiden tulee olla tietoisia suunnittelemiensa rakenteiden kustannuksista jo luonnosvaiheessa ja raportoida tätä tietoa tilaajalle. (Haahtela & Kiiras 2014, s. 27–28)

Hankkeelle pyritään asettamaan alussa realistinen kustannustavoite, jota seurataan suunnittelun edetessä erilaisin menetelmin. Aluksi tilaaja määrittelee tilaohjelman, jossa esitetään tarvittavien tilatyyppien määrä- ja laatutavoitteet. Tämän jälkeen tilaohjelmaan liitetään valittuja ominaisuuksia vastaavat kustannukset ja näin saadaan koko hankkeen kustannusarvio. Tilaohjelmaa ja siihen liitettyä kustannusarviota tutkimalla tilaaja voi muokata tilojen ominaisuudet ja niitä vastaavat kustannukset hyväksyttäväksi kokonaisuudeksi. Tätä lopullista kustannusarviota kutsutaan hankkeen kustannustavoitteeksi. Kustannustavoitteessa huomioidaan usein tilaohjelman lisäksi muun muassa rakennusajankohta, markkinatilanne, hankkeen sijainti sekä laskennan epätarkkuus ja mahdolliset riskit. Esimerkki kustannustavoitteen määrittelystä on esitetty taulukossa 1. (Lindholm 2009, s. 11–13)

Taulukko 1. *Esimerkki kustannustavoitteen määrittelystä (Lindholm 2009, s. 12).*

Tila	m²/tila	kpl	m²	€/m²	€
1h + kk	25	32	800	1900	1 520 000
2h + kk	35	102	3570	2000	7 140 000
2h + k	52	102	5304	2100	11 138 400
3h + k	63	135	8505	2100	17 860 500
4h + k	76	32	2432	2100	5 107 200
Saunaosasto	29	3	87	2500	217 500
Talopesula	20	3	60	2800	168 000
Varastotila	50	3	150	1500	225 000
		Tilat yhteensä (alv 0 %)			**43 376 600**

Kustannustavoitteen määrittelyyn voidaan käyttää erilaisia menetelmiä, kuten esimerkiksi tilalaskentaa, tavoitehinta-, viitekohde-, erokustannus-, tilasto- tai tuotemallimenettelyä. Usein kustannuksia arvioidaan myös eri menetelmiä yhdistelemällä. Yleisesti käytettävässä tilalaskennassa tilahinnat sisältävät kaikki hankkeeseen liittyvät kustannukset, kun taas tavoitehintamenettelyssä otetaan tilalaskennan lisäksi huomioon suhdanteet, hanke- ja tilatekijät. Näitä hankkeen ja tilojen erityispiirteitä ovat esimerkiksi hankekoko, pohjaolosuhteet, rakennuksen vaippa, talotekniikka ja sisäpuoliset pinnat. Viitekohde- ja erokustannusmenettelyssä hyödynnetään aiemmin toteutettuja samankaltaisia hankkeita ja tietoa niiden toteutuneista kustannuksista. Näitä menetelmiä käytettäessä on erityisen tärkeää valita sopiva hanke, johon kustannuksia verrataan. Myös tilastomenettelyssä hyödynnetään jo toteutuneita hankkeita, mutta tällöin vertailukohteita on useita. Tuotemallimenettelyssä kohteesta tehdään heti hankkeen alussa tietomalli, jota hyödynnetään kustannusten arvioinnissa. Tietomallinnuksen kehittyessä tätä menettelytapaa voidaan hyödyntää yhä enemmän. (Lindholm 2009, s. 13–15)

Kun ohjelmavaihe on hankesuunnittelun loputtua saatu päätökseen, tehdään investointipäätös ja siirrytään suunnitteluvaiheeseen. Kohteesta aletaan tuottaa ehdotus- ja luonnossuunnitelmia, minkä seurauksena kustannusten arviointi voidaan liittää suoraan suunniteltuihin rakenteisiin. Suunnitteluvaiheessa kustannusten arviointi perustuu rakennusosalaskentaan, jossa rakennuskustannukset perustuvat suunniteltujen rakennusosien määriin ja niihin liitettäviin yksikköhintoihin. Rakennusosat luokitellaan yleensä yhteisesti sovittujen sääntöjen mukaan, jotta hankkeen eri osapuolien välinen kommunikointi helpottuu. Rakennusosat nimetään ja ryhmitellään yleensä jonkin yleisesti käytetyn nimikkeistön, kuten esimerkiksi Talo 2000:n mukaan, jolloin tietoa voidaan verrata helposti myös muihin hankkeisiin. Lisäksi nimikkeistön mukaisille rakennusosille on määritetty keskimääräiset kustannukset, joita voidaan käyttää hyödyksi kustannuslaskennassa. (Lindholm 2009, s. 15–18)

Erityisesti suunnittelun alkuvaiheessa, jolloin kustannuksiin pystytään vaikuttamaan enemmän, määrä- ja kustannuslaskenta on melko epätarkkaa. Ehdotus- ja luonnossuunnittelussa kaikkien rakennusosien määrät eivät ole vielä tiedossa, mikä aiheuttaa laskentaan epätarkkuutta. Puutuvia määriä arvioidaan lähinnä laskijan ammattitaidon ja kokemusperäisen tiedon avulla, jolloin laskennassa korostuu ihmisten henkilökohtainen osaaminen. Nykyään laskennassa hyödynnetään paljon myös tietomalleja ja muita tietoteknisiä sovelluksia, mikä vähentää laskennan epätarkkuutta. Kun hanke etenee toteutussuunnitteluun, tarkentuvat myös määrä- ja kustannuslaskenta. Erityisesti tietomalleja hyödyntämällä suunniteltavista rakenteista saadaan tarkkaa tietoa suhteellisen helposti. Toteutussuunnittelussa kustannuksiin ei kuitenkaan pystytä enää merkittävästi vaikuttamaan, minkä vuoksi kustannusten kehittymistä lähinnä vain seurataan. (Lindholm 20009, s. 16–17)

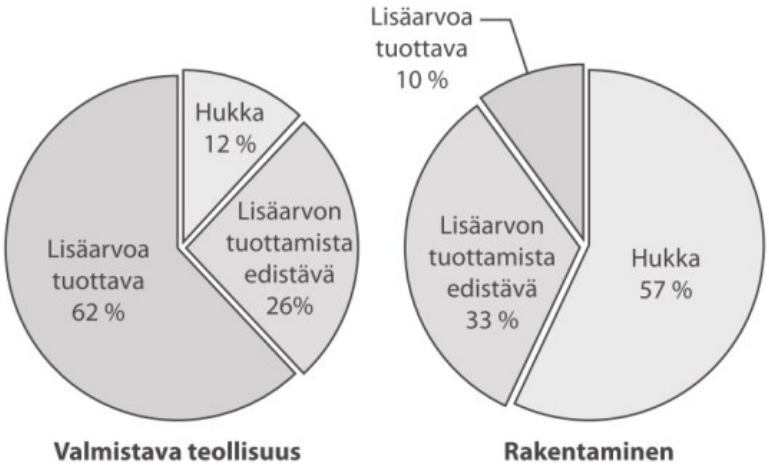
Koska rakennuskustannukset määräytyvät pääasiassa hankkeen alkuvaiheessa, on arkkitehdeilla suuri vastuu suunniteltavan kohteen kokonaiskustannuksista. Myös rakennesuunnittelijat voivat vaikuttaa merkittävästi hankkeen kustannuksiin rakennuksen perustusten, rungon ja muiden täydentävien rakenteiden valinnalla. Arkkitehti määrittelee rakennuksen rungon kannalta melko tiukat määreet, joiden sisällä rakennesuunnittelijan tulee pysyä, mutta myös rakennesuunnittelija pystyy valinnoillaan vaikuttamaan kustannuksiin. Rakennesuunnittelussa pyritään valitsemaan yleisesti käytettyjä rakenneratkaisuja, joiden suunnittelusta ja toteutuksesta on jo aikaisempaa kokemusta. Lisäksi rakennesuunnittelija pyrkii suunnittelussa optimoimaan käytettävät rakennejärjestelmät ja niissä käytettävät rakenteet. Rakenteiden optimointi, kuten mitoitus muutenkin, on iteratiivinen, aikaa vievä prosessi. Siksi rakenteita ja niiden muodostamia kokonaisuuksia ei ole tarkoituksenmukaista optimoida liian pitkään. Optimointi ja täten myös suunnittelun kustannusvaikutusten hallinta perustuu pääasiassa kokeiluun ja kokemusperäiseen tietoon. Siksi suunnittelijoiden henkilökohtaisella ammattitaidolla on suuri vaikutus hankkeen kustannusten määräytymiseen.

2.5 Nykyisen suunnitteluprosessin puutteet ja ongelmakohdat

Rakentamisen Laatu RALA ry on kiinteistö- ja rakennusalan toimija, joka kerää palautetta erilaisista rakennushankkeista niiden toteutukseen osallistuvilta yrityksiltä. Junnosen ja Kärnän vuonna 2015 julkaistussa suunnittelijapalautteen analyysissä tarkastellaan suunnittelijoiden saamia sekä heidän muille rakennushankkeeseen osallistuville tahoille antamia palautteita. Palautteiden mukaan suunnittelijoiden ja muiden rakennushakkeen osapuolien välinen yhteistyö toimii pääasiallisesti hyvin, mutta myös ongelmakohtia löytyy. Suunnittelijat saavat muilta osapuolilta kritiikkiä erityisesti suunnitelmien virheellisyydestä ja ristiriitaisuudesta sekä huonosta kustannustietämyksestä. Lisäksi puutteita nähdään suunnittelutyön ajallisessa hallinnassa sekä suunnittelijoiden keskinäisessä yhteistyössä. Suunnittelijoiden näkökulmasta ongelmallisiksi asioiksi luetellaan muun

muassa puutteelliset lähtötiedot, projektin huono ajallinen hallinta sekä heikko suunnittelun ohjaus.

Rakennushankkeiden suunnitteluun ja toteutukseen osallistuu nykyään paljon eri toimijoita, jolloin niiden välisen yhteistyön tärkeys korostuu. Koskenvesan (2010) mukaan eri osapuolien välisen yhteistyön puute on ehkä suurin yksittäinen rakennusten suunnittelua ja koko rakennushankkeen läpivientiä vaivaava ongelma. Nykyrakentaminen on jakautunut moniin pieniin osakokonaisuuksiin, mikä tekee siitä monimutkaista. Hankkeen eri osapuolet ohjaavat projektia omien etujensa mukaiseen suuntaan, mikä vähentää hankekokonaisuuden tuottavuutta sekä aiheuttaa ongelmia ja ristiriitoja. Tällöin rakentamisprosesseissa syntyy paljon hukkaa, mikä ei tuota asiakkaalle mitään lisäarvoa. Rakennusalalla syntyvän hukan määrä on selvästi suurempi muihin teollisuuden aloihin verrattuna, minkä johdosta tähän asiaan tulee kiinnittää erityistä huomiota. Asiakkaalle lisäarvoa tuottavien toimintojen osuuksia esitellään kuvassa 12.



Kuva 12. *Lisäarvoa asiakkaalle tuottavien toimintojen osuus valmistavassa teollisuudessa ja rakentamisessa (Construction Institute USA 2004, Koskenvesa 2010 mukaan).*

Suunnittelijoiden välisen yhteistyön tulisi olla tiivistä, jotta kohteen suunnittelu voidaan toteuttaa tehokkaasti. Kun on kyse eri toimijoiden välisestä yhteistyöstä, pitää jonkun ottaa vastuu suunnittelunohjauksesta ja varmistaa näin suunnittelijoiden välisen yhteistyön toimivuus. Yleensä suunnittelunohjauksesta vastaa pääsuunnittelija sekä rakennuttajakonsultti, mutta myös tietomallikoordinaattori voi osallistua ohjaukseen nykyajan tietomallinnettavissa hankkeissa. Suunnittelunohjauskäytännöissä koetaan olevan suuria puutteita, vaikka se on yksi tärkeimmistä rakennushankkeen onnistumiseen vaikuttavista tekijöistä. Tämä johtaa siihen, että suunnittelijoilla ei ole riittävästi tietoa esimerkiksi hankkeen tavoitteista, aikataulusta tai budjetista. Suunnittelunohjauksen käytännöt ovat nykyisellään puutteellisia eikä todellista, nykyään käytettyä suunnitteluprosessia ole kuvattu. Tämän vuoksi projektinjohto usein ohjaa suunnittelua epärealististen tavoitteiden,

aikataulujen ja budjetin pohjalta. Suunnittelunohjaus ei ole puutteellista ainoastaan hanketasolla vaan ongelmia on myös eri suunnittelualojen sisällä. Projektien johtamisessa ja esimerkiksi suunnittelun aikataulutuksessa koetaan olevan monia puutteita, minkä vuoksi asetettuihin tavoitteisiin ei päästä. (Lassila 2016, s. 46–47)

Rakennushankkeiden aikataulut venyvät usein huonon kommunikaation, epäselvien vaatimusten ja yleisten väärinkäsitysten vuoksi (Kerosuo et al. 2012). Lassilan (2016) mukaan suunnitteluaikataulujen sisältö ja seuranta koetaan puutteelliseksi, mikä johtaa usein aikataulujen viivästymiseen. Suunnittelutehtävät aikataulutetaan yleisellä tasolla, mutta tämä tarkkuustaso ei ole riittävä. Suunnittelutehtävistä ilmoitetaan yleensä niihin tarvittava aika, mutta eri tehtävien välisten riippuvuussuhteiden ja niiden muodostaman kokonaisuuden suunnittelu laiminlyödään. Tämän vuoksi suunnittelijoiden on vaikea ymmärtää toisten osapuolien tehtävien sisältöä ja niiden lähtötietovaatimuksia ja siksi aikataulussa pysyminen vaikeutuu. Lähtökohtaisesti virhe on siinä, että kaikkien osapuolien suunnitelmille on usein määritetty sama valmistumisaika, vaikka esimerkiksi rakennesuunnittelija tarvitsee arkkitehdin suunnitelmia lähtötiedokseen.

Yleisesti ottaen rakennushankkeiden aikataulut ovat yleensä kireitä ja epärealistisia. Aikataulut eivät ole myöskään joustavia vaan, jos suunnittelussa ilmenee ongelmia tai siihen tulee muutoksia, joudutaan uudet tehtävät tyypillisesti sovittamaan alkuperäiseen aikatauluun. Suunnittelusta on vaikeaa löytää selviä rajakohtia, jolloin eri tehtävät kasaantuvat päällekkäin ja aikataulu sekoittuu. Tehtävien päällekkäisyyden vuoksi monet suunnittelijat kokevat muutosepävarmuutta, ja tämän vuoksi pyrkivät venyttämään päätöksenteon lähelle sovittua valmistumispäivämäärää. Tämän avulla suunnittelija minimoi suurten muutosten tekemisen omiin suunnitelmiinsa, mutta kokonaisuus kärsii. Tämän vuoksi onkin hyvin tärkeää, että suunnitteluaikataulu jaksotetaan riittävän pieniin osiin. Kun suunnittelun aikataulusta tehdään kireä, vaikuttaa se usein suunnittelun laatuun ja täten myös rakennuskustannuksiin negatiivisesti. (Lassila 2016)

Yksi syy suunnittelun huonoon aikataulutukseen ovat vanhentuneet, perinteiset keinot, joita käytetään yhä tietomallipohjaisiksi muuttuneissa rakennushankkeissa. Tietomallinnus tuo paljon hyötyjä, mutta se myös hidastaa joitakin suunnitteluvaiheita selvästi. Esimerkiksi muutosten tekeminen tietomalliin koetaan usein työlääksi ja hitaaksi, koska mallia joudutaan purkamaan ja rakentamaan uudelleen. Tietomallipohjainen suunnittelu onkin usein perinteisiä menetelmiä hitaampaa, mutta tämä johtuu pääasiassa suunnittelutiedon määrän ja laadun kasvusta. Myös ohjelmistojen puutteet ja käyttäjien osaamattomuus aiheuttavat ongelmia. Tietomallinnus saattaa tuoda huomattavasti lisää työtä suunnittelijoille, jolloin on tärkeää miettiä eri mallinnustöiden tuoman lisäarvon määrää. Kaikki työt on tärkeää sitoa asetettuihin tavoitteisiin, jotta ylimääräiseltä työltä vältytään. (Lassila 2016, s. 38–40)

Tietomallinnuksessa nähdään yleisesti paljon hyötyjä ja potentiaalia, mutta sitä ei osata vielä kaikilta osilta täysin hyödyntää. Yleiset toimintatavat eivät ole vielä muotoutuneet

tietomallinnettaviin hankkeisiin, mikä aiheuttaa ongelmia ja aikataulujen venymistä. Tilaajat eivät myöskään tiedä tarkalleen tietomallintamisesta saatavia hyötyjä, jolloin tavoitteiden asettelu jää usein puutteelliseksi. Tietomallit mahdollistavat monien avustavien toimintojen tehostamisen, kuten piirustusten tuotannon nopeuttamisen, mutta hyötyjen tarkastelussa tulee aina muistaa kokonaisuus. Esimerkiksi rakenteiden mallintaminen saattaa olla hyödyllistä piirustusten teon kannalta, mutta mallinnuksen tarkkuustaso tulee tutkia erikseen. (Lassila 2016, s. 48–50)

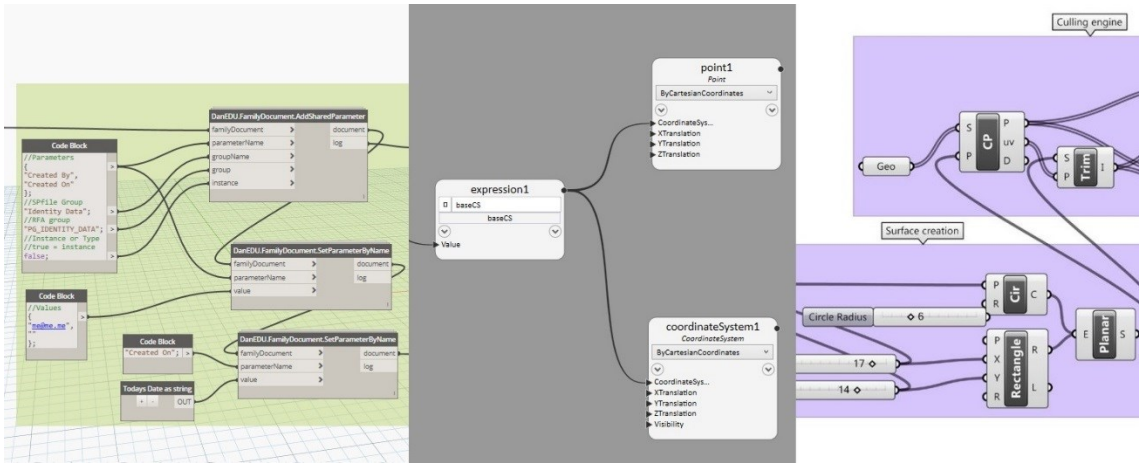
Yksi merkittävä suunnittelussa havaittu ongelma, joka haittaa erityisesti suunnitteluosapuolien välistä yhteistyötä, on tiedonsiirron vaikeus. Ongelmana on suunnittelijoiden käyttämät erilaiset ohjelmistot, jotka käyttävät erilaisia tiedosto- ja tallennusmuotoja. Kun tietoa siirretään suunnitteluohjelmistojen välillä, saattaa tärkeää tietoa hävitä matkalla. Myös ohjelmistojen erilaiset käyttötavat saattavat hankaloittaa suunnitelmien tulkintaa ja hyödynnettävyyttä. (RT 10-10992, s. 5) Sekä hankeosapuolien että yksittäisen suunnittelijan käyttämien suunnitteluohjelmistojen välinen tiedonsiirto tapahtuu pääasiassa manuaalisesti. Koska manuaalinen tiedonsiirto on tavallisesti melko hidasta, ei sitä yleensä tehdä kuin tarvittaessa. Tämän vuoksi suunnittelutietoa ei esimerkiksi jaeta muille suunnitteluosapuolille kuin tietyin, ennalta päätetyin aikavälein. Nopeasti etenevässä suunnittelussa tämä on ongelma, koska suunnittelijoilla ei ole päivitettyjä lähtötietoja käytössään. Lisäksi siirrettävän tiedon hyödynnettävyys on yleisesti melko alhaisella tasolla. Esimerkiksi rakennesuunnittelija voi käyttää arkkitehdin muodostamaa IFC-mallia lähinnä vain pohjana mallinnukselle eli arkkitehdin kertaalleen mallinnetut rakenteet joudutaan mallintamaan uudelleen rakennemalliin.

Kaikki edellä mainitut suunnitteluprosessin puutteet ja ongelmat johtavat lopulta hankeen kokonaiskustannuksiin. Yleisesti ajatellaan, että muutosten lisääntyessä myös kustannukset lisääntyvät. Erityisesti suunnitteluvaiheessa tapahtuvat muutokset vaikuttavat kustannuksiin merkittävästi. Tilaajan ja häntä mahdollisesti ohjaavan rakennuttajakonsultin tulee olla tietoisia erilaisten suunnitteluratkaisujen kustannuksista. Ennen kaikkea suunnittelun alkuvaiheessa tulee tehdä päätöksiä, jotka ovat realistisia eivätkä aiheuta tarpeetonta lisäsuunnittelua. Ongelmaksi koetaankin suunnittelijoiden huono kustannustietämys, minkä johdosta heidän omien suunnitelmiensa kustannusseurantaa pidetään puutteellisena. Myös kustannusseurannassa tulee aina olla mukana tavoite, mitä kohti kuljetaan. Yleensä ongelmana on, että kustannustavoite on olemassa, mutta se tai sen välitavoitteet eivät ole suunnittelijoiden tiedossa. (Lassila 2016, s. 52)

3. ALGORITMIAVUSTEINEN SUUNNITTELU

Algoritmiavusteista suunnittelua hyödynnetään vielä suhteellisen vähän rakennusalalla. Yksi osoitus tästä on se, ettei algoritmiavusteiselle suunnittelulle ole muodostunut vakiintunutta termistöä. Aiheesta on kirjoitettu vasta hyvin vähän suomenkielistä kirjallisuutta, minkä vuoksi yleisesti käytettyä käsitteistöä ei ole vielä ehtinyt syntyä. Englanninkielistä aineistoa löytyy jo melko paljon, mutta niissä samoja asioita kuvaavat termit vaihtelevat laajasti eri lähteiden välillä. (Tanska & Österlund 2014, s. 18)

Tanskan ja Österlundin (2014) mukaan algoritmiavusteinen suunnittelu on algoritmeja hyödyntävää suunnittelua, jossa jokin suunnittelun osa-alue on ratkaistu algoritmisen prosessin avulla. Tämä algoritminen prosessi voi olla joko manuaalinen, ilman tietokonetta tapahtuva, tarkasti määritelty tehtäväsarja tai tietokoneen laskentatehoa hyödyntävä ohjelmoitu sarja. Nykyajan tietoteknisessä ympäristössä algoritmit luodaan ja suoritetaan pääasiassa aina tietokonetta hyväksi käyttäen. Rakennusalalla algoritmiavusteinen suunnittelu tapahtuu pääasiassa erilaisten visuaalisten ohjelmointialustojen avulla. Yleisesti käytettyjä ohjelmia ovat esimerkiksi Autodeskin Dynamo, Bentleyn Generative Components ja McNeelin Grasshopper. Kuvassa 13 on esitetty otteet edellä mainittujen ohjelmien käyttöliittymistä.



Kuva 13. *Visuaaliset ohjelmointialustat Dynamo, Generative Components ja Grasshopper (Dynamo Forum; Bentley Communities; Davis 2013).*

Algoritmiavusteiseen suunnitteluun liitetään usein myös käsite parametrinen mallinnus. Perinteisen tietomallinnuksen (BIM) voi myös katsoa olevan parametrista mallinnusta, koska siinä luodaan ja muokataan mallinnusobjekteja erilaisten parametrien avulla. Mallinnetut rakenteet toimivat ohjelmoituina objekteina, jotka muuttuvat niihin liitettyjä parametreja muutettaessa. Perinteisessä tietomallinnuksessa suunnittelija kuitenkin mallintaa vain rakenteet, ja käytetty ohjelma generoi omilla sisäisillä algoritmeillaan rakentei-

den älykkäitä ominaisuuksia ja niiden välisiä suhteita. Algoritmiavusteisessa suunnittelussa lähestymistapa on lähes päinvastainen perinteisiin menetelmiin verrattuna. Siinä keskitytään erityisesti erilaisten suunnittelutietokokonaisuuksien älykkääseen yhdistelyyn ja toiminnallisen tietoverkon luomiseen. Suunnittelutieto on hyvin yksinkertaista geometria- ja metatietoa, jota luodaan, muokataan ja yhdistellään erilaisten matemaattisten sääntöjen ja ehtojen avulla. Algoritmiavusteisten suunnittelumenetelmien avulla voidaan näin siirtyä yksittäisistä parametrisista objekteista kokonaiseen parametrisesti muokattavaan malliin. (Boeykens 2012, s. 1–2)

Tässä luvussa tutkitaan algoritmiavusteista suunnittelua erityisesti rakennesuunnittelun näkökulmasta. Luvussa tutkitaan algoritmiavusteisten menetelmien ominaisuuksia ja tarkastellaan niiden soveltuvuutta rakennusten suunnitteluprosessiin. Ensin käydään läpi algoritmien ja niiden muodostamien prosessien taustalla vaikuttavaa teoriaa sekä algoritmiavusteisen suunnittelun taustaa. Seuraavaksi käsitellään algoritmiavusteisen suunnitteluprosessin ominaisuuksia, suunnittelussa yleisesti käytettävää visuaalista ohjelmointialustaa Grasshopperia sekä algoritmimallin luomista, muokkaamista ja analysointia. Lopuksi tarkastellaan vielä algoritmiavusteisen suunnittelun hyötyjä ja haasteita.

3.1 Teoria ja tausta

Koska tämän tutkimuksen aihepiiristä on vasta hyvin vähän suomenkielistä kirjallisuutta, ei aiheesta ole syntynyt vielä yleisesti käytettyä termistöä. Monet termit ovatkin suoria käännöksiä vaihtelevista englanninkielisistä käsitteistä, minkä seurauksena käytettävien termien valitseminen on vaikeaa. Tämän työn pääaiheesta käytetään termiä algoritmiavusteinen suunnittelu, englanniksi Algorithm Aided Design (AAD). Tämän termin synonyyminä voidaan käyttää myös käsitteitä algoritminen suunnittelu tai parametrinen suunnittelu. Parametrisella suunnittelulla (engl. parametric design) viitataan kirjallisuudessa joskus myös perinteiseen suunnittelutapaan, mutta tässä työssä parametrisuudella viitataan aina algoritmeja suoraan hyödyntävään menetelmään. Muutkin termit muodostuvat tyypillisesti perinteisten termien pohjalta lisäämällä niiden eteen algoritmiavusteinen, algoritminen tai parametrinen. Esimerkiksi algoritmiavusteisessa suunnitteluprosessissa suoritettavasta tietomallinnuksesta voidaan käyttää termiä parametrinen mallinnus.

Algoritmiavusteisen suunnittelun ja siinä käytettävän visuaalisen ohjelmoinnin voidaan katsoa alkaneen GRAIL-systeemin myötä jo vuonna 1969. Nykyisessä muodossaan sitä alettiin kuitenkin käyttää vasta 2000-luvun alussa, kun Generative Components tuli markkinoille vuonna 2003 ja Grasshopper vuonna 2007. (Davis et al. 2011, s. 363) Tällä hetkellä algoritmiavusteista suunnittelua hyödynnetään pääasiassa arkkitehtisuunnittelussa, mutta viime vuosina algoritmisia suunnittelumenetelmiä on alettu käyttää myös rakennusteknisissä sovellutuksissa. Todellisissa rakennushankkeissa parametrisia menetelmiä on rakennesuunnittelun osalta käytetty vielä melko vähän, mutta joitakin kohteita löytyy. Tunnettuja kohteita, joissa algoritmiavusteista suunnittelua on käytetty laajamittaisesti ovat esimerkiksi Macaulle rakennettava Morpheus-hotelli, Perthiin rakennettu Elizabeth

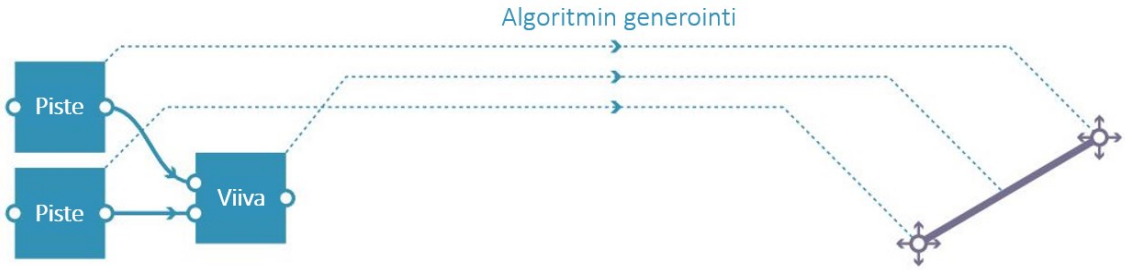
Quayn kevyen liikenteen silta sekä Dubliniin rakennettu, kuvassa 14 esitetty, Aviva Stadium.

Kuva 14. *Aviva Stadium (Scott Tallon Walker Architects).*

Algoritmiavusteista suunnittelua voidaan pitää osana tietokoneavusteisen suunnittelun (engl. Computer-Aided Design, CAD) kolmatta sukupolvea. Aish esittelee vuonna 2013 julkaistussa artikkelissa tietokoneavusteisen suunnittelun kolme sukupolvea: 2D-piirto, (engl. 2D drafting), tietomallinnus (BIM) ja laskentaa hyödyntävä suunnittelu (engl. design computation). Kolmannen sukupolven suunnittelumenetelmät keskittyvät rakenteita kuvaavien algoritmien ja niiden avulla generoitavien mallien luomiseen. Tällöin suunnittelija ei mallinna rakenteita vaan luo algoritmien avulla säännöt, joiden avulla ohjelma osaa mallintaa rakenteet automatisoidusti. Näin tietokoneen suuri laskentateho saadaan hyödynnettyä paremmin ja suunnittelun tehokkuus paranee. Kolmannen sukupolven suunnittelumenetelmät muuttavat myös suunnittelutiedon sisällön ja rakenteen. Tämä uudenlainen tietorakenne mahdollistaa kevyemmin prosessoitavien mallien rakentamisen ja niiden hyödyntämisen erilaisissa simuloinneissa ja optimoinneissa. Tämä on suuri etu perinteisiin, tietosisällöltään suuriin ja raskaisiin tietomalleihin verrattuna.

Rakennusten algoritminen suunnittelu tapahtuu pääasiassa visuaalisten ohjelmointialustojen avulla. Nämä visuaaliset ohjelmointiympäristöt voivat olla joko itsenäisiä ohjelmia tai toimia jonkin BIM-ohjelman lisäosana. Esimerkiksi yleisesti graafiseen ohjelmointiin käytettävä Grasshopper on integroitu McNeelin Rhinoceros 3D -ohjelmaan (Rhino) siten, että Grasshopperissa luodut algoritmit ohjaavat Rhinossa esitettävää tietomallia. Visuaalinen ohjelmointi mahdollistaa ohjelmoinnin graafisten elementtien avulla ilman, että käyttäjän tarvitsee syöttää tekstimuotoista koodia. Sen avulla voidaan luoda täysin samalla tavalla toimivia, sääntöihin perustuvia riippuvuussuhteita ja operaatioita kuin tekstimuotoisella ohjelmoinnillakin.

Visuaalinen ohjelmointi koostuu tekstimuotoisen koodin sijasta visuaalisista objekteista, joiden avulla muodostetaan suunnittelussa tarvittavat algoritmit. Visuaaliset ohjelmointialgoritmit muodostuvat solmupisteistä (engl. node) ja niitä yhdistävistä linkeistä (engl. link). Solmupisteet suorittavat aina jonkin niille määritetyn tehtävän ja niitä voidaan karkeasti verrata yhteen riviin tekstimuotoista koodia. Todellisuudessa solmupisteinä toimivat komponentit sisältävät useita rivejä koodia, mikä mahdollistaa komponenttien oikeanlaisen toiminnan. Solmupisteitä yhdistävien linkkien avulla voidaan määrittää solmupisteiden suorittamien tehtävien järjestys ja muodostaa näin tietoa muokkaavat algoritmit. Kuvassa 15 on esitetty yksinkertainen esimerkki, jossa luodaan visuaalisen ohjelmoinnin avulla kaksi pistettä ja niiden välille viiva. (Davis 2013)



Kuva 15. *Periaatekuva visuaalisesta ohjelmoinnista perustuen lähteeseen (Davis 2013, s. 102).*

Solmupisteinä toimivat komponentit ja niitä yhdistävät linkit muodostavat kokonaisuuksia, joita voidaan kutsua algoritmeiksi. International Electrotechnical Commission (IEC) (2013) määrittelee algoritmin olevan täysin määritelty äärellinen sarja ohjeita, joiden avulla syötteestä (input) voidaan muodostaa tuloste (output). Algoritmien rakennetta ja niiden osatekijöitä voidaan luokitella eri tavoin. Woodbury (2010) jakaa solmupisteet kolmeen kategoriaan perustuen niiden paikkaan algoritmissa. Osa solmupisteistä on niin sanottuja lähdepisteitä (engl. source node), jotka aloittavat algoritmeja. Nämä pisteet eivät siis vastaanota dataa ollenkaan vaan toimivat aina algoritmin lähtötietona. Algoritmin aloittavia lähdepisteitä kutsutaan usein myös parametreiksi. Ne ovat tyypillisesti numeroarvoja tai geometriatietoja, kuten pisteitä, viivoja tai pintoja. Seuraava solmupistetyyppi on algoritmin sisäinen piste (engl. internal node). Ne toimivat algoritmin tiedonkulussa niin sanottuina välittäjäpisteinä, jotka vastaanottavat syötteen ja välittävät tulosteen eteenpäin. Viimeinen pistetyyppi on lopetuspiste (engl. sink node), joka ainoastaan vastaanottaa tietoa, mutta ei välitä sitä eteenpäin. Lyhyimmillään algoritmi voi koostua ainoastaan yhdestä solmupisteestä, jolloin tämä kyseinen piste toimii sekä algoritmin alku- että loppupisteenä.

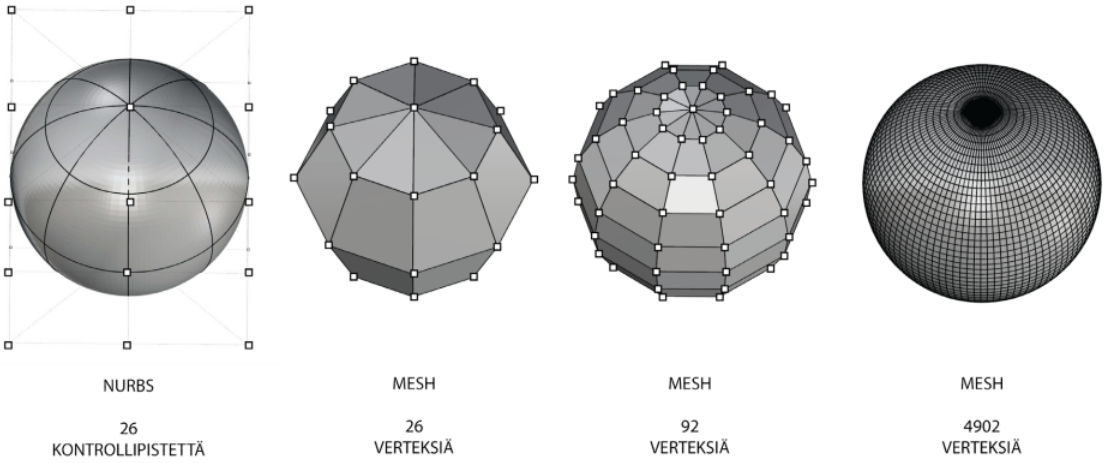
Visuaalisen ohjelmoinnin periaate on käyttää yksinkertaisessa muodossa olevaa tietosisältöä, jotta visuaalinen ohjelmointialusta pystyy prosessoimaan nopeasti malliin tehtäviä muutoksia. Yksi visuaalisen ohjelmoinnin perusta onkin mallin parametrisoiminen ja nopea muokkaaminen parametreja muuttamalla. Todellisuudessa visuaalisessa ohjelmoinnissa solmupisteiden ja niitä yhdistävien linkkien avulla muodostetut ohjelmat käyvät läpi

kolme erityyppistä algoritmia toimiakseen oikein. Nämä algoritmit mahdollistavat visuaalisen ohjelmointialustan ja sen ohjaaman 3D-mallin toimimisen. Algoritmit ovat osa visuaalisen ohjelmoinnin toimintaperiaatetta eikä suunnittelija voi suorannaisesti vaikuttaa niiden toimintaan. (Woodbury 2010)

Ensimmäinen algoritmi määrittää visuaalisen ohjelman suoritusjärjestyksen. Se etsii solmupisteiden suoritusjärjestyksen siten, että solmupisteet prosessoivat ja välittävät tietoa eteenpäin vasta, kun kaikki tarvittavat syötteet on vastaanotettu. Visuaalisessa ohjelmointialustassa luodut algoritmit eivät voi muodostaa tietokehiä. Jos jonkin solmupisteen suorittama tehtävä vaatii lähtötiedokseen sen oman tulosteen, ei algoritmi pysty suorittamaan sille määritettyä tehtäväsarjaa. Kun suoritusjärjestys on määritetty, toinen algoritmi käy läpi kaikki solmupisteet ja prosessoi niihin ohjelmoidut tehtäväkuvaukset. Hyvin toimivissa järjestelmissä tämä algoritmi tarkastelee ainoastaan solmuja, joiden lähtötiedot ovat muuttuneet ja siten nopeuttaa ohjelman toimintaa. (Woodbury 2010, s. 15–16)

Kolmannen algoritmin tehtävänä on luoda sekä visuaalisesta kuvaajasta että sen muodostamasta 3D-mallista graafinen esitys. Aina kun suunnittelija tekee muutoksia lähtötietoina toimiviin parametreihin tai mallia muokkaaviin komponentteihin, ohjelmointialusta suorittaa nämä kolme algoritmia. Jos malli ei ole suhteettoman suuri, ohjelma pystyy käsittelemään algoritmit sekunnin murto-osissa. Tällöin visuaalisen ohjelmointialustan muokkaaman 3D-esityksen voi kokea päivittyvän reaaliaikaisesti. Mitä suuremmaksi mallin tietosisältö kasvaa ja mitä monimutkaisemmilla tavoilla se on toteutettu, sitä kauemmin ohjelmistolta kestää prosessoida malliin tehdyt muutokset. (Woodbury 2010, s. 15–16)

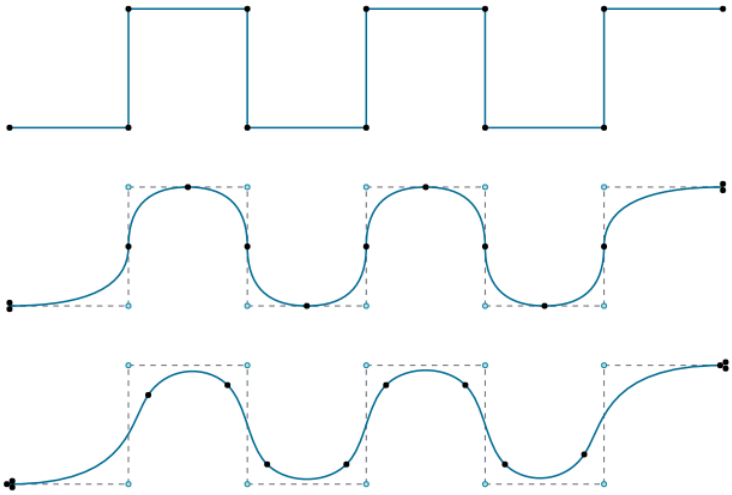
Yksi merkittävä algoritmiavusteisen tietosisällön ominaisuus on NURBS. NURBS eli Non-Uniform Rational Basis Spline on geometrian mallinnus- ja esitystapa, jonka avulla voidaan luoda tarkkoja, matemaattisesti määritettyjä käyriä, pintoja ja kolmiulotteisia geometrioita. NURBS on vaihtoehtoinen geometriamuoto esimerkiksi mesh-geometrialle, jossa geometria muodostuu useista mesh-elementeistä. Koska NURBS-geometria perustuu matemaattisiin kaavoihin, on se aina jokaisessa pisteessään mittatarkkaa ja tältä osin myös täsmällisempää kuin mesh-geometria. Myös mesh-pohjaisella geometrialla voidaan saavuttaa tarkkoja malleja, mutta tämä vaatii hyvin tiheiden ja raskaiden elementtiverkkojen käyttöä. NURBS- ja mesh-geometrian eroa on havainnollistettu kuvassa 16. (Tanska & Österlund 2014)



Kuva 16. *Kolmiulotteinen pallo NURBS-pinnalla ja erilaisilla mesh-pinnoilla kuvattuna (Tanska & Österlund, s. 30).*

NURBS-geometria voidaan määritellä neljän eri tekijän avulla: geometrian aste (degree), kontrollipisteet (control points), solmut (knots) ja arviointisääntö (evaluation rule). Geometrian aste on pääasiassa 1, 2, 3 tai 5, mutta teoriassa se voi olla mikä tahansa positiivinen kokonaisluku. Asteluku kuvaa kontrollipisteiden vaikutusta geometrian muodostumiseen ja siksi asteluvun pienentäminen vaikuttaa geometrian muotoon. Toisaalta asteluvun kasvattaminen ei välttämättä muuta olemassa olevaa geometriaa. Kontrollipisteitä käytetään pääasiassa geometrian muokkaamiseen ja niiden lukumäärän tulee aina olla vähintään asteluku + 1. Kontrollipisteillä on positiivisella numerolla kuvattava ominaisuus nimeltään paino (weight). Se on tyypillisesti kaikilla kontrollipisteillä sama, mutta rationaalisilla geometrioilla, kuten NURBS-geometrioilla, kontrollipisteiden painoluku voi myös vaihdella. (Rhinoceros 3D)

Kolmantena NURBS-geometriaa määrittävänä tekijänä ovat solmut, jotka muodostavat listan numeroita eli niin sanotun solmuvektorin (knot vector). Solmuvektorin lukuja muuttamalla voidaan vaikuttaa kontrollipisteitä ohjaavien solmujen määrään ja sijaintiin ja siten erityisesti kaarevien muotojen muodostumiseen. Solmupisteistä muodostuva lukujono voi olla joko homogeeninen tai epähomogeeninen. Kuten NURBS nimikin kuvaa, sillä määritetty geometria voi olla myös epähomogeenista. Tämä tarkoittaa käytännössä sitä, että geometrian kuvaus on joustavampaa kuin homogeenista geometriaa käytettäessä. Kuvassa 17 on esitetty erään käyrän muodostuminen erilaisia solmuvektoreita käyttämällä. Viimeinen geometriaa määrittelevä tekijä on niin sanottu arviointisääntö. Se on matemaattinen kaava, joka ottaa huomioon geometrian asteen, kontrollipisteet ja solmut. Lisäksi arviointisääntö sisältää NURBS-geometrian B-käyrien (basis spline) perusfunktiot. (Rhinoceros 3D)



Kuva 17. *Käyrän kontrollipisteiden ohjaaminen erilaisten solmuvektoreiden avulla (Grasshopper Primer 2014, s. 77).*

3.2 Algoritmiavusteinen suunnitteluprosessi

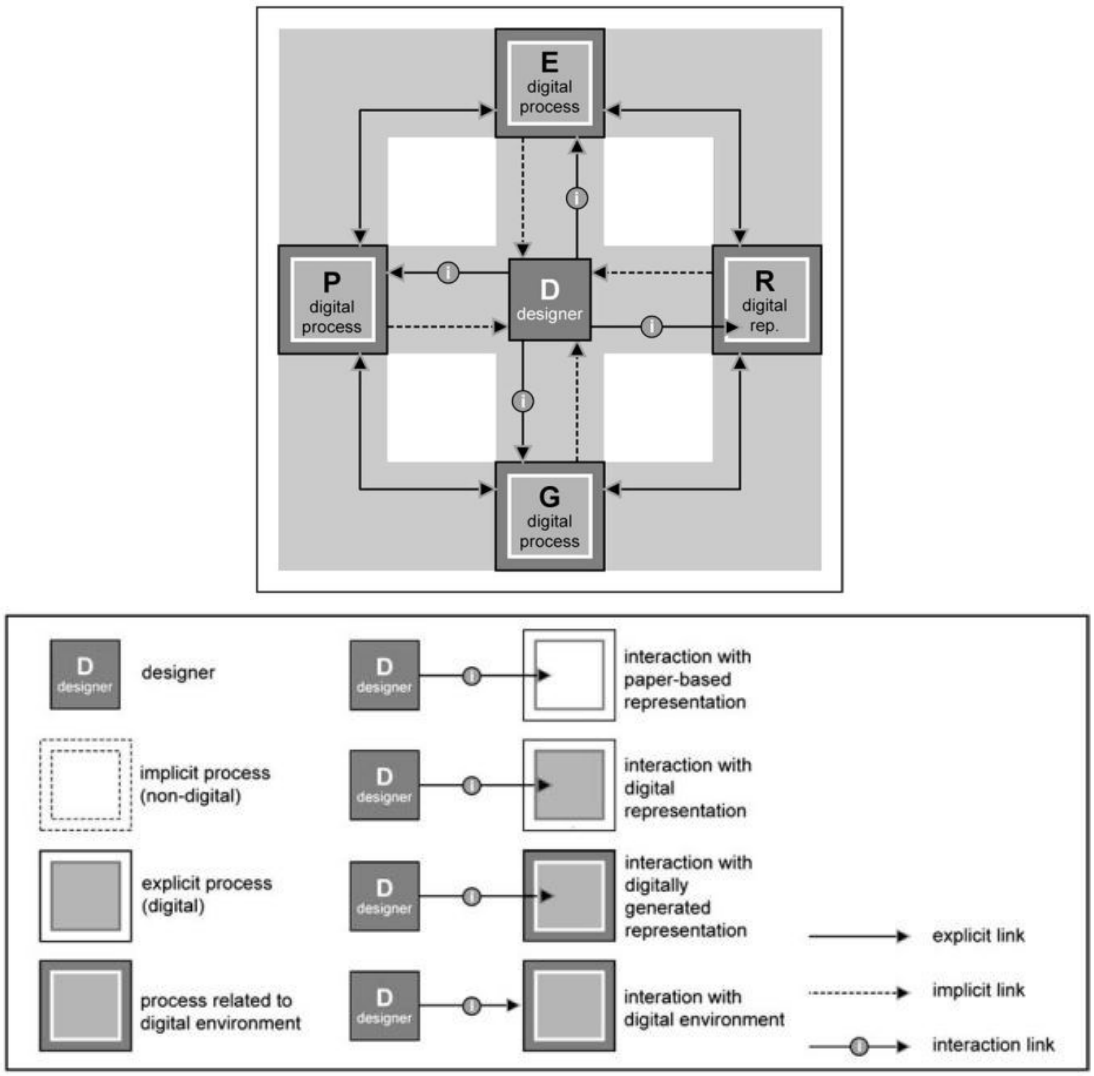
Algoritmiavusteisessa suunnitteluprosessissa on tarkoitus suunnitella rakenteita algoritmisia työkaluja hyödyntämällä. Suunnittelun tavoite on sama kuin perinteisiä menetelmiä käytettäessä, mutta siihen johtava prosessi eroaa perinteisestä. Erilaisten suunnittelutyökalujen käyttö ei itsessään tarkoita suunnitteluprosessin muuttumista, mutta algoritmisten menetelmien kohdalla myös suunnitteluprosessin voidaan katsoa muuttuvan. Algoritmiavusteisen suunnitteluprosessin suurin ero perinteiseen suunnitteluun verrattuna on eri työvaiheiden ja -tehtävien linkittymien toisiinsa ohjelmoinnin avulla.

Kuten luvussa 2 esitetään, suunnittelu voidaan jakaa kolmessa eri vaiheessa tapahtuvaan suunnittelukokonaisuuteen: rakennejärjestelmiin, rakenneosiin ja liitoksiin. Algoritmiavusteisen suunnitteluprosessin etenemiseen vaikuttaa ennen kaikkea algoritmisten suunnittelumenetelmien käyttölaajuus, sillä niitä on mahdollista hyödyntää kaikilla kolmella tasolla, yhdessä tai erikseen. Prosessiin vaikuttavat myös muiden suunnitteluosapuolien käyttämät menetelmät ja heiltä saatavien lähtötietojen ominaisuudet. Tässä työssä keskitytään tarkastelemaan prosessia, jossa käytetään algoritmiavusteisia menetelmiä koko suunnittelukokonaisuuden ajan, aina rakennejärjestelmien suunnittelusta liitosten suunnitteluun. Lisäksi tarkasteltavassa prosessissa arkkitehdilta saatavat lähtötiedot ovat algoritmiavusteisesti tuotettuja. Algoritmiavusteista suunnittelua on mahdollista hyödyntää niin mitoitukseen, mallinnukseen kuin raportointiinkin.

3.2.1 Algoritmiavusteisen suunnitteluprosessin ominaisuudet

Oxman esittää vuonna 2006 julkaistussa artikkelissa viisi erilaista kuvausta suunnitteluprosessin ominaisuuksista, joista yhdistelmämalli kuvaa hyvin nykyisin käytettävää algoritmiavusteista suunnittelua. Yhdistelmämallilla tarkoitetaan suunnitteluprosessia, jossa

kaikki suunnittelun osatekijät on integroitu toisiinsa. Tällöin suunnittelu koostuu yhtenäisestä systeemistä, jossa mallin luominen (G), sen esitysmuoto (R), sisällön arviointi ja analysointi (E) sekä mallinnettujen rakenteiden suorituskyky (P) toimivat yhtenäisenä, toisensa huomioon ottavana kokonaisuutena. Erityisesti algoritmiavusteiset menetelmät mahdollistavat tällaisen systeemin käyttämisen. (Oxman 2006) Oxmanin esitys suunnitteluprosessin yhdistelmämallista on esitetty kuvassa 18.



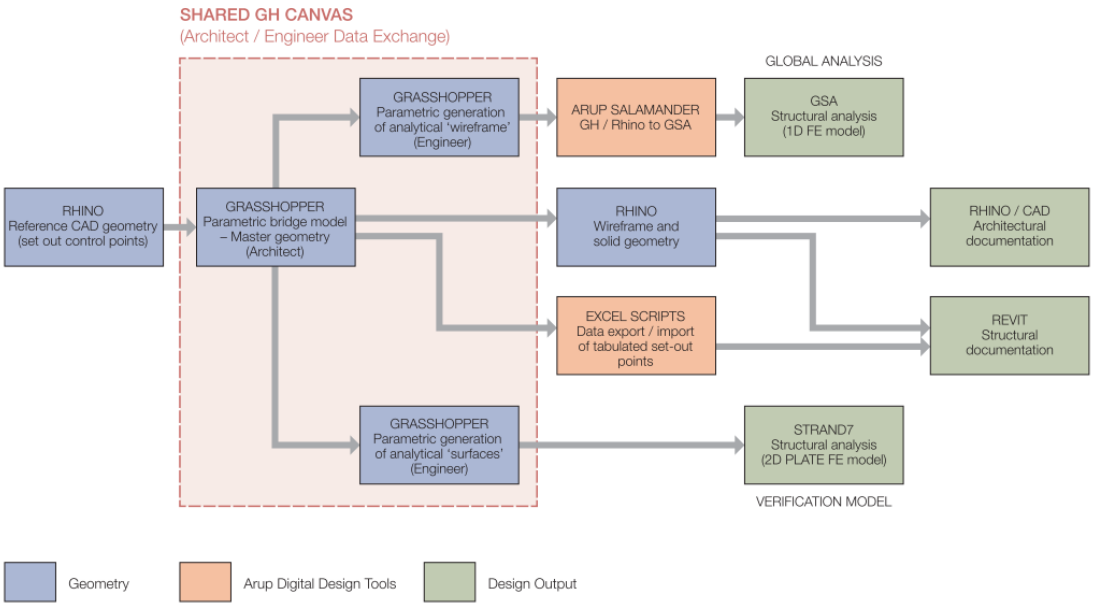
Kuva 18. *Algoritmiavusteisen suunnitteluprosessin ominaisuudet (Oxman 2006).*

Edellä esitetyssä kuvassa 18 kuvataan suunnittelijan ja suunnittelun eri osatekijöiden välisiä suhteita sekä osaprosessien ominaisuuksia. Kaavion komponentit ovat samat kuin perinteisessäkin suunnitteluprosessissa, mutta osatekijöiden ja niitä yhdistävien linkityksien ominaisuudet ovat muuttuneet. Myös algoritmiavusteisessa suunnittelussa tulee edelleen mitoittaa, mallintaa ja raportoida suunniteltavat rakenteet, mutta kaikissa näissä tehtävissä voidaan hyödyntää algoritmisia menetelmiä.

Algoritmiavusteisen suunnitteluprosessin voi katsoa noudattavan samoja periaatteita kuin suunnittelussa hyödynnettävän visuaalisen ohjelmoinnin. Visuaalisessa ohjelmoinnissa

algoritmit koostuvat komponenteista ja niitä yhdistävistä linkeistä, kun taas suunnitteluprosessi muodostuu erilaisista malleista ja muista tietorakenteista sekä niitä yhdistävistä rajapinnoista. Algoritmiavusteista suunnitteluprosessia voi kuvata eräänlaisena ekosysteeminä, joka sisältää monia perinteisiä tietomuotoja, kuten esimerkiksi taulukoita, tietomalleja ja laskentamalleja sekä niitä ohjaavia ja yhdistäviä algoritmeja. Prosessi koostuu usein erillisistä, visuaalisen ohjelmointialustan avulla rakennetuista algoritmeja sisältävistä malleista, jotka keräävät ja käsittelevät lähtötietoina saatavaa suunnittelutietoa ja muodostavat siitä muissa malleissa hyödynnettävää tietosisältöä. Näitä malleista käytetään tässä työssä nimitystä algoritmimalli. Kun kaikki suunnitteluun tarvittavat tiedot ovat linkitettyinä toisiinsa erilaisten älykkäiden rajapintojen avulla, voi koko suunnitteluprosessia pitää yhtenäisenä, muutoksiin automaattisesti reagoivana systeeminä.

Algoritmiavusteisessa suunnittelussa hyödynnetään edelleen samoja ohjelmistoja ja tiedostomuotoja kuin perinteisessäkin suunnitteluprosessissa. Esimerkiksi yksinkertaista suunnittelutietoa voidaan yhä tarkastella taulukkolaskentaohjelmassa, tietomallia perinteisessä BIM-ohjelmassa ja laskentamallia FEM-ohjelmassa. Koska kaikki suunnittelutieto kuitenkin kuvaa samaa suunnittelukohdetta, on esimerkiksi geometriaa turha rakentaa erikseen useisiin eri malleihin. Jokin algoritmimalli voikin esimerkiksi sisältää ohjeet suunniteltavan rakennuksen rautalankamallin luomiseen, mitä voidaan hyödyntää sekä tieto- että laskentamallin päägeometrian lähtötietona. Näin ollen molempien mallien muokkaaminen onnistuu samanaikaisesti, yhtä algoritmimallia päivittämällä. Kuvassa 19 esitetään esimerkki algoritmiavusteisen suunnitteluprosessin etenemisestä. Kyseessä on Australian Perthiin rakennetun kevyen liikenteen sillan suunnitteluprosessi, jossa hyödynnettiin laaja-alaisesti algoritmiavusteisia suunnittelumenetelmiä sekä arkkitehdin että rakennesuunnittelijan toimesta.



Kuva 19. *Tiedonkulku algoritmiavusteisessa suunnitteluprosessissa (Avern-Taplin et al. 2016).*

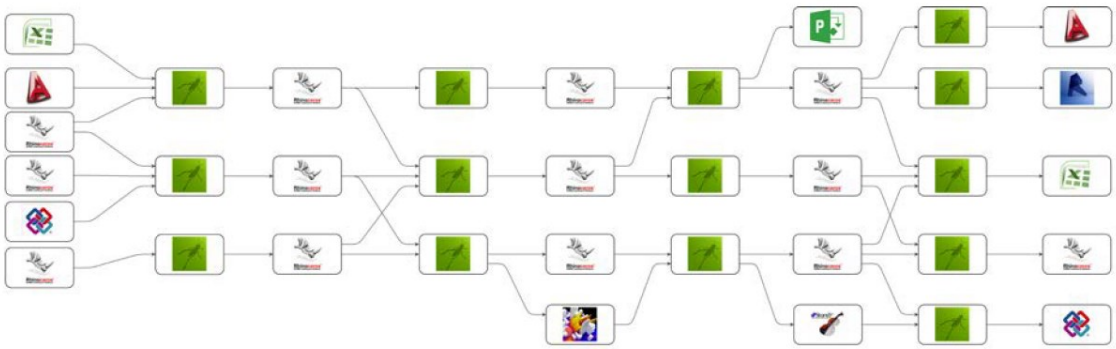
Boeykensin (2012) mukaan algoritmiavusteisessa suunnitteluprosessissa on järkevää hyödyntää myös perinteisen tietomallintamisen vahvuuksia. Hänen mukaansa perinteinen tietomalli on myös algoritmisessa prosessissa hyvä työkalu esimerkiksi suunnittelun raportointiin ja suunniteltavien rakenteiden havainnollistamiseen. Perinteisestä prosessista eroten tietomalli luodaan ja muokataan algoritmiavusteisessa prosessissa erilaisten sääntöjen ja algoritmien sekä muun ulkoisen informaation avulla. Näin ollen suunnitteluprosessin keskiössä eivät ole enää niinkään käytettävät työkalut ja ohjelmistot vaan informaation kulku eri käyttäjien, ohjelmistojen ja tiedostojen välillä. Suunnitteluprosessissa käytettävät erilaiset työkalut ja ohjelmat voidaan tällöin yhdistää tiiviiksi verkostoksi, jossa kukin ohjelma toimii kuitenkin itsenäisesti, toisista ohjelmista riippumattomasti. Tämä ohjelmistoista riippumaton tiedonkulku voidaan toteuttaa joko neutraalien siirtotiedostomuotojen tai ohjelmien välisten linkityksien avulla. Yksi mahdollisuus on myös luoda ohjelmistoja, jotka sisältävät sekä mallin generoimiseen käytettävät säännöt että tuotetun mallin tietosisällön havainnollistamisen.

Perinteisessä BIM-mallintamisessa malli luodaan tyypillisesti sijoittamalla valmiita mallinnusobjekteja 3D-ympäristöön ja päivittämällä näiden objektien sisältämää suunnittelutietoa suunnittelun edetessä. Koska algoritmiavusteisessa suunnittelussa tietomallintaminen eroaa huomattavasti perinteisestä BIM:stä, Van der Heijden et al. (2015) esittelevät julkaisussaan termin Building Information Generation (BIG). BIG kuvaa ennen kaikkea tapaa, jolla tietomalli luodaan. Se on logiikkaa sisältävä prosessi, jonka avulla suunnitteluprosessissa luotavat yksinkertaiset tietorakenteet kerätään ja yhdistetään toisiinsa. Lisäksi suunnitteludataan yhdistetään erilaisia ominaisuuksia, jotka mahdollistavat esimerkiksi erilaisten rakenneosien luokittelun. Algoritmiavusteisessa suunnittelussa tietomallin luomisen voidaan katsoa noudattavan BIG-prosessin periaatteita.

Algoritmiavusteisessa suunnittelussa käytettävän BIG-metodologian avulla luotavat mallit perustuvat niin sanotun yksinkertaisen ja kevyen suunnittelutiedon käyttöön. Suunnittelussa hyödynnettävä data voidaan jakaa karkeasti kahteen osaan: geometriatietoon ja siihen liitettävään ominaisuustietoon. Yksinkertaistettuja geometriamalleja voidaan kutsua myös nimellä rautalankamalli (engl. wireframe model), sillä ne koostuvat yksinkertaisista geometrisista komponenteista, kuten esimerkiksi pisteistä ja viivoista. BIG alkaakin tyypillisesti geometriatiedon luomisella, minkä jälkeen siihen aletaan liittää erilaisia ominaisuuksia. Geometriatietoon linkitettävät ominaisuudet sisältävät aina jonkin tunnuksen ja tunnukseen liitettävän arvon. Ominaisuudet kuvaavat geometriakomponentteja sekä luovat linkityksiä erilaisten tietorakenteiden välille. BIG-prosessissa on tavoitteena pitää myös ominaisuustiedot yksinkertaisessa helposti prosessoitavassa muodossa. Tämä voidaan saavuttaa esimerkiksi ylläpitämällä erillistä tekstitiedostoa, taulukkoa tai tietokantaa, johon ominaisuustiedot on koottu. Ominaisuustieto voidaan myös sisällyttää suoraan käytettävään geometriaan. (Van der Heijden et al. 2015, s. 419)

BIG-prosessi voidaan jakaa vaiheisiin, jotka toimivat kuin suunnittelutietoa muokkaavat funktiot. Suunnitteluprosessin eri vaiheilla on niihin tulevat lähtötiedot, lähtötietoja

muokkaavat tehtävät sekä prosessivaiheesta eteenpäin jatkavat tulokset. Suunnitteluprosessi eteneekin täysin samalla periaatteella kuin visuaalinen ohjelmointi. Jotta suunnitteluprosessin eri vaiheet voidaan liittää tarkoituksenmukaisesti toisiinsa, ominaisuustietojen tulee olla määritettynä riittävän kattavasti. Ominaisuustietojen avulla voidaankin esimerkiksi suodattaa, järjestää ja lajitella geometriakomponentteja ja näin muokata suunnitteludataa paremmin hyödynnettävään muotoon. Suunnitteluprosessin eri vaiheissa luodaan monia pienempiä malleja, joita tavallisesti hyödynnetään jossakin suunnittelua edistävässä sovellutuksessa. Vaikka näistä prosessin eri vaiheissa luoduista malleista muodostetaan suunnittelun edetessä monimutkaisempia yhdistelmämalleja, myös aikaisempien vaiheiden mallit ovat hyödynnettävissä koko suunnitteluprosessin ajan. (Van der Heijden 2015, s. 419–420) Kuvassa 20 esitetään algoritmiavusteisessa suunnittelussa käytettävien osaprosessien muodostama suunnittelukokonaisuus.



Kuva 20. *Erilaisten mallien ja prosessivaiheiden muodostama algoritmiavusteinen suunnitteluprosessi perustuen lähteeseen (Van der Heijden 2015, s. 420).*

Kuten kuvassa 20 esitetystä kaaviosta nähdään, prosessissa luotavat mallit perustuvat lähes aina johonkin toiseen malliin. Eri mallien väliset linkitykset tuleekin rakentaa niin, että ne toimivat aina myös prosessin eri vaiheissa luotujen mallien muuttuessa. Linkityksien avulla prosessi voidaan jakaa useisiin pieniin malleihin, minkä ansiosta mallien tiedostokoko pysyy pienempänä ja niitä muokkaavat säännöt yksinkertaisempina. Tällöin myös suunnitelmamuutosten prosessointi on huomattavasti nopeampaa kuin perinteisten, suurikokoisten BIM-mallien kohdalla. (Van der Heijden 2015, s. 420)

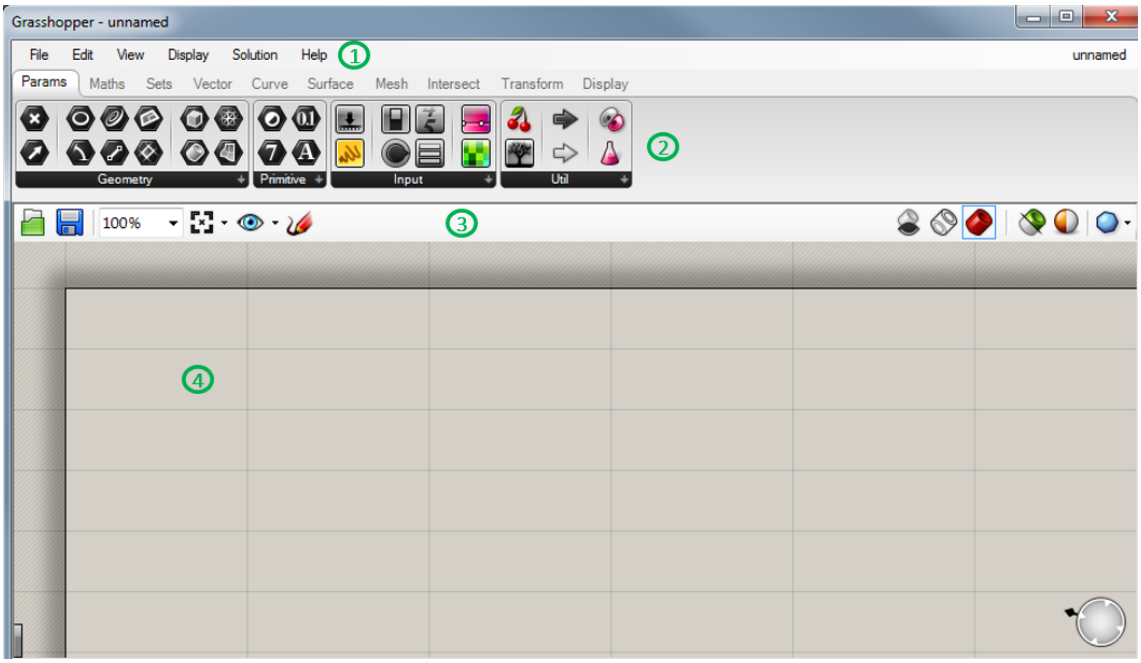
Perinteisessä suunnitteluprosessissa luodusta BIM-mallista poiketen, BIG-prosessissa tuotettavaa mallia ei aloiteta lähes koskaan täysin puhtaalta pöydältä. Vaikka perinteisessäkin prosessissa saadaan usein erilaisia tiedostoja uuden mallin lähtötiedoksi, niitä harvoin pystytään hyödyntämään suoraan mallinnustyössä. BIG-prosessissa luotavien mallien lähtötietoina voidaan hyödyntää hyvin monenlaisia tiedostoja, kuten esimerkiksi taulukoita, kuvaajia, tietokantoja, piirustuksia ja 3D-malleja. Mallin rakentaminen alkaakin aina saatavilla olevien lähtötietojen kartoituksella (engl. data mapping) ja näiden tietojen käsittelyllä ja yhdistämisellä. Näin voidaan minimoida uuden mallin rakentamiseen käytettävä aika ja samalla vältytään eri mallien välisiltä ristiriitaisuuksilta. (Van der Heijden 2015, s. 420)

Kun lähtötiedot on kerätty ja käsitelty, voidaan niistä koota uusi malli. Tämän jälkeen mallia muokataan ja prosessoidaan erilaisten funktioiden avulla ja samalla tyypillisesti kasvatetaan suunnittelukokonaisuuden tietosisältöä. Mallin muokkauksessa tulee muistaa prosessoida samanaikaisesti sekä geometriaa että siihen liitettyä ominaisuustietoa. Jos esimerkiksi geometrinen osa jaetaan kahtia, tulee myös osaan liitetyn tietosisällön päivittyä oikein uusiin osiin. Kun suunnittelutieto on prosessoitu tarvittavien funktioiden avulla, voidaan uutta mallia hyödyntää erilaisiin suunnittelua tukeviin tarkoituksiin. Mallista voidaan generoida monenlaisia tuloksia, kuten esimerkiksi tietomalleja, tietokantoja sekä kuvia ja kuvaajia. (Van der Heijden 2015, s. 421–422)

3.2.2 Grasshopper

Algoritmiavusteinen suunnittelu tapahtuu pääasiassa visuaalisen ohjelmointialustan avulla. Yksi visuaaliseen ohjelmointiin yleisimmin käytettävä ohjelmisto on David Ruttenin kehittämä Grasshopper. Se on graafinen algoritmieditori, joka toimii yhdessä Rhinoceros 3D -mallinnusohjelman (Rhino) kanssa. Grasshopper on Rhinossa käytettävä lisäosa, jonka avulla voidaan luoda parametrisesti ohjattavia ja automaattisesti päivittyviä malleja. Grasshopperissa luodut algoritmit ohjaavat niihin ohjelmoitujen sääntöjen avulla Rhinossa tapahtuvaa mallinnustyötä. Vaikka Grasshopper toimii aina yhdessä Rhinon kanssa, voidaan se linkittää myös moniin muihin ohjelmistoihin. Grasshopper on luonteeltaan hyvin avoin ohjelmisto, minkä ansiosta sen muokkaaminen ja laajentamien on suhteellisen helppoa. Grasshopperiin on mahdollista ohjelmoida omia komponentteja ja lisäosia, jotka mahdollistavat ohjelman monipuolisen käytön. Grasshopperiin onkin tarjolla paljon erilaisia Grasshopper-käyttäjien kehittämiä lisäosia, joiden avulla sen toiminta-alaa voidaan laajentaa. (Grasshopper Primer 2014, s. 3–8)

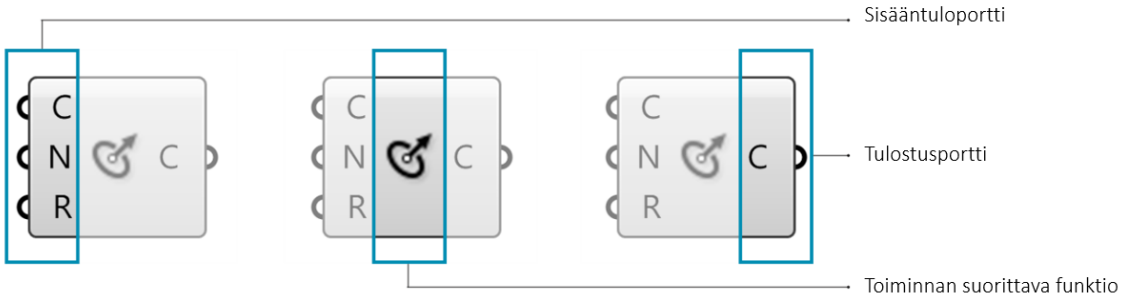
Grasshopper voidaan avata syöttämällä Rhinon komentoriville komento ”Grasshopper”, minkä jälkeen ohjelma avautuu omaan ikkunaansa. Grasshopperin käyttöliittymä on hyvin yksinkertainen. Se koostuu neljästä peruskomponentista, jotka ovat: päävalikko (1), komponenttivalikko (2), kangastyökalurivi (3) sekä kangas (4). Käyttöliittymä ja sen osat on esitetty kuvassa 21. Kenties tärkein käyttöliittymän osa on komponenttivalikko, koska se sisältää kaikki ohjelman sisältämät komponentit. Jotta halutunlaisen komponentin löytäminen olisi helpompaa, on komponentit jaettu niiden tyypin mukaan eri valikoihin ja alavalikoihin. Myös kaikki Grasshopperiin ladatut lisäosat ja niiden sisältämät komponentit löytyvät tyypillisesti uuden välilehden alta komponenttivalikosta. Komponenttia voi etsiä myös hakutoiminnon avulla tuplaklikkaamalla kangasta ja kirjoittamalla aukeavaan ikkunaan hakusanan. (Grasshopper Primer 2014, s. 3–8)



Kuva 21. *Grasshopperin käyttöliittymä.*

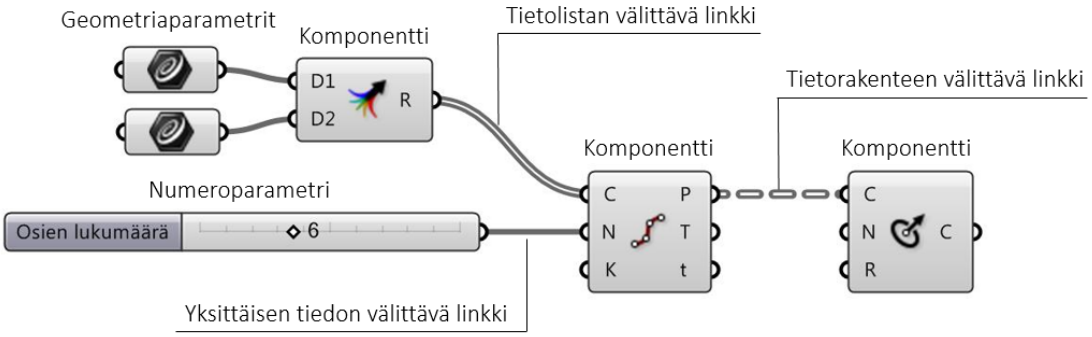
Grasshopperin kankaalle luotavat algoritmit koostuvat solmupisteinä toimivista parametreista ja komponenteista sekä niitä yhdistävistä linkeistä. Parametrit sisältävät tietoa, jota komponentit käsittelevät ja muokkaavat. Parametrien sisältämä tieto on joko vaihtelevaa (engl. volatile) tai pysyvää (engl. persistent) ja se voi olla tyypiltään esimerkiksi numeroita, värejä tai geometriatietoa. Pysyvä tieto on käyttäjän erikseen määrittämä arvo, jonka on tarkoitus pysyä muuttumattomana. Esimerkiksi geometrista pistettä kuvaava parametri on pysyvää tietoa, koska kyseinen parametri on määritetty kuvaamaan yhtä tiettyä pistettä. Vaihtelevaa tietoa ei ole määritetty pysyvästi vaan se muuttuu aina edeltävien määritelmien muuttuessa. Suurin osa parametreista sisältää tyypillisesti muuttuvaa tietoa. (Grasshopper Primer 2014, s. 43–44)

Komponentit ovat objekteja, jotka suorittavat erilaisia tehtäviä niihin ohjelmoitujen sääntöjen perusteella. Ne voidaan tyypillisesti jakaa kolmeen toiminnalliseen osaan. Ensimmäinen osa on niin sanottu syöttöportti (engl. input port), jonka kautta tieto tulee komponenttiin. Näitä syöttöportteja voi olla useita ja ne voivat vastaanottaa erilaisia syötteitä. Toinen osa on komponenttiin ohjelmoitu funktio, joka käsittelee syötettä siihen ohjelmoidulla tavalla. Viimeinen osa on niin sanottu tulostusportti (engl. output port), josta käsitelty tieto voidaan johtaa eteenpäin. Myös tulostusportteja voi olla useampia ja ne voivat antaa erimuotoisia tulosteita. (Janssen & Chen 2011, s. 801–802; Grasshopper Primer 2014, s. 36) Grasshopperissa käytettävien komponenttien pääosat on esitetty kuvassa 22.



Kuva 22. *Visuaalisen ohjelmakomponentin pääosat perustuen lähteeseen (Grasshopper Primer 2014, s. 38).*

Erilaisia parametreja ja komponentteja voidaan yhdistellä toisiinsa linkkien avulla. Linkit kuvaavat tiedon kulkua objektien välillä ja esittävät näin algoritmien läpi kulkevan tiedon reitin (Janssen & Chen 2011, s. 801). Jos tietoa ei ole määritetty pysyvässä muodossa suoraan komponenttiin, pitää se tuoda jostakin linkityksiä hyväksi käyttäen. Linkkien avulla voidaan yhdistää vain toisiinsa sopivia sisääntulo- ja tulostusportteja tai muuten ohjelma ilmoittaa ongelmasta. Grasshopperissa luotavat linkitykset eivät kuitenkaan ole yhtä tarkasti määritettyjä kuin esimerkiksi tekstimuotoisen ohjelmoinnin kohdalla. Grasshopperin komponentit osaavatkin tehdä monia parametrimuunnoksia automaattisesti, jolloin käyttäjän työ helpottuu. Komponenttien väliset linkit eivät kerro ainoastaan suunnittelutiedon etenemisreittiä vaan ne voivat myös esittää linkin läpi kulkevan tiedon ominaisuuksia. Käytännössä Grasshopper sisältää kolme erilaista linkin esitystapaa. Yksi harmaa viiva kuvastaa yksittäistä tietoyksikköä, kaksoisviiva tietolistaa ja katkoviiva puumaista tietorakennetta. Nämä erilaiset esitysmuodot helpottavat ennen kaikkea monimutkaisten algoritmien tulkintaa. (Grasshopper Primer 2014, s. 48–49). Grasshopperissa luodun algoritmin pääosat on esitetty kuvassa 23.



Kuva 23. *Visuaalinen ohjelmointialgoritmi ja sen osat perustuen lähteeseen (Grasshopper Primer 2014, s. 49).*

Valmiiden ohjelmakomponenttien lisäksi Grasshopperissa voi käyttää myös omia tekstimuotoisesti ohjelmoituja komponentteja. Komponentteja voi luoda suoraan Grasshopperissa yksinkertaisten valmiiden työkalujen avulla. Yleisimmin käytetyt ohjelmointikielet ovat C#, Visual Basic sekä Python (Grasshopper Primer 2014, s. 8). Suunnittelijan itse määrittämien komponenttien visuaalinen ilme ja toimintaperiaate ovat usein täysin samanlaiset kuin valmiiden komponenttien, mutta niiden suorittama tehtävä on ohjelmoitu itse. Yhteen komponenttiin voidaan tarvittaessa ohjelmoida hyvinkin monimutkainen, monta tekstiriviä sisältävä tehtäväsarja, jonka toteuttamiseen valmiiden komponenttien ominaisuudet eivät riitä.

3.2.3 Algoritmimallin luominen

Tietomallintamiseen laadun ja hyödynnettävyyden varmistamiseksi alan toimijat ovat yhteistyössä luoneet tietomallintamisessa käytettävän ohjeistuksen, yleiset tietomallivaatimukset. Lisäksi koska BIM-mallit ovat nykyään hyvin laajasti käytetty suunnittelutyökalu, on alan yrityksille muodostunut kehittyneet, vakioituneet tavat luoda tietomalleja. Algoritmiavusteisesti luotavien tietomallien kohdalla voidaan hyödyntää pitkälti samoja ohjeistuksia, vaikka mallien luominen eroaakin huomattavasti perinteisestä mallintamisprosessista. Algoritmiavusteisessa suunnittelussa käytettävien algoritmimallien kohdalla näitä tapoja ja ohjeistuksia ei voida kuitenkaan hyödyntää. Myös algoritmimallien kohdalla on tärkeää noudattaa hyvää mallinnustapaa, jotta mallien käytettävyys ja hyödynnettävyys voidaan maksimoida. Algoritmeja luotaessa on kiinnitettävä erityisesti huomiota tiedon jaotteluun, ryhmittelyyn, nimeämiseen ja numerointiin sekä parametrien määrittelyyn.

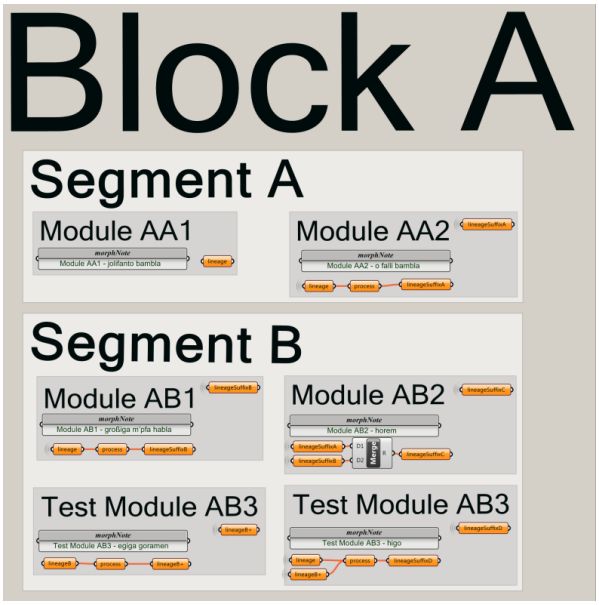
Jotta visuaalisen ohjelmoinnin avulla tuotettuja algoritmimalleja voidaan hyödyntää mahdollisimman laaja-alaisesti suunnittelussa, on niiden rakenteeseen kiinnitettävä erityistä huomiota. Algoritmien luokittelun ja ryhmittelyn voi katsoa olevan oma suunnittelutehtävänsä, joka on hyvin tärkeä suunnittelutiedon hallinnoin kannalta (Magnusson et al. 2017). Zboinska (2015) käy julkaisussaan läpi, miten visuaalisten ohjelmointialgoritmien muodostamat kokonaisuudet tulisi organisoida tehokkaasti. Hänen tavoitteenaan on löytää erityisesti tekijät, joiden avulla algoritmien luettavuus ja ymmärrettävyys paranevat. Zboinska tarkastelee visuaalisen ohjelmoinnin tehokkuuden parantamista erityisesti arkkitehtisuunnittelussa, mutta samoja periaatteita voidaan hyödyntää myös rakennesuunnittelun tehtäväkentässä.

Suunnittelijoiden luomista algoritmimalleista voi helposti muodostua hyvin vaikeaselkoisia, jolloin niiden ulkopuolinen tarkastelu on vaikeaa. Kun visuaalisen ohjelmoinnin käyttö yleistyy, on tärkeää kiinnittää huomiota algoritmimallien selkeään rakenteeseen. Tällöin suunnittelussa aika ei kulu algoritmien tutkimiseen ja niiden logiikan opetteluun vaan parametrisen suunnittelun edut saadaan täysin hyödynnettyä. Tutkimuksessaan Zboinska (2015) esittää kahdeksanosaisen yleisohjeen visuaalisten ohjelmointialgoritmien jäsentelyyn, hallintaan ja optimointiin:

1. Käytä algoritmeja ja havainnollisia kuvia.
2. Jaottele ja jäsentele algoritmeja eri tasoilla.
3. Käytä toistuvia suunnittelukokonaisuuksia.
4. Luetteloi, numeroi ja nimeä kokonaisuudet ja niiden osat.
5. Jaottele ja järjestä käytettävät parametrit.
6. Ryhmittele osakokonaisuudet selkeiksi kokonaisuuksiksi.
7. Käytä värikoodausta sekä algoritmi- että parametriryhmissä.
8. Yksinkertaista ja optimoi algoritmit.

Edellä esitetyt ohjeistukset soveltuvat tutkimuksen mukaan useisiin erityyppisiin algoritmeihin ja niiden muodostamiin algoritmimalleihin. Ohjeiden koettiin soveltuvan hyvin kaikkiin tutkittuihin algoritmityyppeihin: geometrian luomiseen, laskennalliseen analyysiin, fysiikkasimulaatioon sekä optimointiin.

Algoritmien tietosisältöä on hyvä selventää esimerkiksi havainnollistavien kuvien avulla. Visuaalinen ohjelmointialusta mahdollistaa kuvallisten kommenttien liittämisen algoritmien yhteyteen. Kun algoritmien ja niiden osien yhteyteen liitetään kuvakaappaus kulloinkin generoitavista 3D-elementeistä, selkeytyy algoritmien toiminta huomattavasti. Yksi tärkeimmistä ja samalla myös haastavimmista tekijöistä algoritmimallin tietosisällön hallitsemisessa on jakaa sitä erilaisiksi kokonaisuuksiksi ja osakokonaisuuksiksi. Erilaiset suunnittelutehtävät, kuten esimerkiksi geometrian luominen ja laskenta-analyysi, on hyvä jakaa omiksi kokonaisuuksikseen. Tietosisällöltään suurissa malleissa nämä kokonaisuudet on hyvä jaotella vielä pienempiin osiin usealla eri tasolla. (Zboinska 2015) Magnusson et al. (2017) jaottelevat ja ryhmittelevät Grasshopperissa luotavat algoritmit kolmella eri tasolla, jotka ovat laajimmasta suppeimpaan: lohko (engl. block), segmentti (engl. segment) ja moduuli (engl. module). Esimerkki algoritmien jaottelusta on esitetty kuvassa 24.



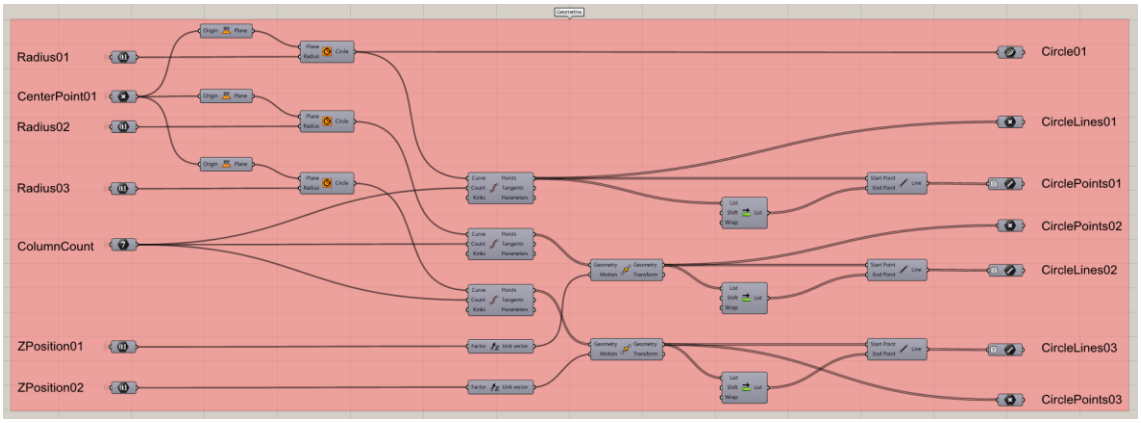
Kuva 24. *Esimerkki algoritmin tietosisällön jaottelusta (Magnusson et al. 2017).*

Tärkeä osa algoritmien jaottelua on sen osien järjestäminen, nimeäminen ja numeroiminen. Erityisesti nimeäminen mahdollistaa algoritmin eri osien selkeämmän yhdistelyn ja algoritmimallin sisällön raportoinnin. Algoritmimallia luotaessa on kiinnitettävä huomiota koko mallin, sen osakokonaisuuksien sekä yksittäisten komponenttien nimeämiseen ja numerointiin. Erityisesti mallin lähtötietona toimivien parametrien selkeä nimeäminen on tärkeää. Nimien tulee olla johdonmukaisia ja selkeitä sekä riittävän lyhyitä, jotta niiden käyttö ja muistaminen on helppoa. Nimien tarkoituksena on helpottaa mallin rakenteen havainnollistamista ja täten mahdollistaa käytettävyys myös tulevaisuudessa. Mallin osien tarkoituksenmukainen nimeäminen helpottaa huomattavasti myös muiden mallia käyttävien osapuolien työtä. Nimeäminen jatkuu koko suunnitteluprosessin ajan nimeämisen, havainnoinnin ja uudelleen nimeämisen muodossa. Kun algoritmimallilla ja sen osilla on selkeät nimet, lisääntyy algoritmimallin hyödynnettävyys ja luotettavuus. (Woodbury 2010, s. 190)

Yksi suurimmista algoritmiavusteisen suunnittelun vahvuuksista on suunnittelutiedon parametrisuus ja siten myös helppo muokattavuus. Näin ollen suunnitteluparametrien määrittely ja hallinnointi ovat yksi merkittävä osa algoritmimallin luomisessa ja muokkaamisessa. Parametrit voidaan jakaa pysyvään (engl. static) ja muuttuvaan (engl. dynamic) tietosisältöön ja tätä jakoa on hyvä käyttää myös algoritmimallia luotaessa. Pysyvät parametrit ovat suuruudeltaan vakioita ja niitä muokataan ainoastaan ulkoisten prosessien avulla. Muuttuvien parametrien arvo voi vaihdella ja niitä muokataan sisäisillä prosesseilla. Tästä syystä parametrit on tärkeää jaotella selvästi ominaisuutensa mukaan. Parametrien tietosisällön pysyvyyden lisäksi ne voidaan jakaa kahteen osaan datan määrän mukaan. Parametri voi sisältää joko yksittäisen attribuuttitiedon (engl. field) tai koostua useita yksittäisiä attribuutteja sisältävästä kokoelmasta (engl. record). Yksittäinen parametri voi olla esimerkiksi yksinkertainen geometriatieto, numero tai väri. Parametrit on hyvä jakaa ryhmiin esimerkiksi niiden avulla generoitavan geometrian mukaan. Kun parametrit on jaoteltu ja ryhmitelty selvästi, suunnittelijan on helpompi havaita, mitä osaa mallissa parametrit ohjaavat ja mitkä parametrit ovat suunnittelijan muokattavissa. (Zboinska 2015, s. 482)

Jotta algoritmimallin yleisilme olisi mahdollisimman havainnollinen, on visuaalisessa ohjelmoinnissa järkevää hyödyntää ryhmien jäsentelyä, värikoodausta ja optimointia. Erilaiset osakokonaisuudet kannattaa ryhmitellä aina siten, että kokonaisuuden syötteet on eritelty vasempaan ja tulosteet oikeaan reunaan. Tämä voidaan toteuttaa käytännössä kahdentamalla ryhmän vastaanottamat syötteet ja tuottamat tulosteet niiden perusmuodossa ryhmän reunoilla. Erilaiset kokonaisuudet on hyvä myös värikoodata niiden ominaisuuksien ja mallinnustason mukaan. Tämä lisää mallin havainnollisuutta huomattavasti. Viimeisiä algoritmimallin rakenteen parantamiseen liittyviä tekijöitä on algoritmien optimointi. Se tarkoittaa käytännössä sitä, että algoritmeista poistetaan tai piilotetaan muokkaamisen kannalta merkityksettömät osat. Piilottaminen voidaan toteuttaa Grasshopperissa luomalla ohjelman sisäisiä ryhmiä (engl. cluster). (Zboinska 2015, s. 482–483) Yksi

esimerkki Grasshopperissa luodun komponenttiryhmän jäsentelystä on esitetty kuvassa 25.



Kuva 25. *Esimerkki suunnittelutiedon ryhmittelystä moduulin sisällä.*

Visuaalisia algoritmeja voidaan yksinkertaistaa myös käyttämällä tekstimuotoista ohjelmointia. Koska valmiiden visuaalisten komponenttien määrä on rajallinen, voi jonkin tehtävän määrittäminen olla tehokkaampaa tekstimuotoisella koodilla. Kun useiden komponenttien vaatima tehtävä voidaan kuvata yhdellä itse koodatulla komponentilla, mallin visuaalinen ilme yksinkertaistuu ja näin ollen mallin tulkinta helpottuu. Toisaalta suunnittelijan luomien, tekstimuotoista koodia sisältävien komponenttien nimeäminen ja niiden toiminnan raportointi on tärkeää, jotta muutkin mallia tarkastelevat henkilöt ymmärtävät algoritmien toimintaperiaatteen.

3.3 Algoritmiavusteinen optimointi

Yksi algoritmiavusteisen suunnittelun vahvuuksista on rakenteiden vaivattomampi optimointi. Algoritmiset menetelmät mahdollistavat kokonaisten rakennejärjestelmien optimoinnin yksittäisten rakenneosien sijasta. Tämän ansiosta rakenteita voidaan optimoida ja kustannusvaikutuksia seurata entistä varhaisemmassa vaiheessa suunnittelua. Brown et al. (2016) käyvät tutkimuksessaan läpi, mitä erilaisia tavoitteita rakennusten suunnittelulla on ja miten kaikki tavoitteet saavutetaan mahdollisimman tehokkaasti. Tutkimuksessa tarkastellaan useaan päämäärään tähtäävää optimointia, josta Brown et al. käyttävät termiä Multi Objective Optimization (MOO). MOO:n tarkoituksena on auttaa suunnitteluryhmää valitsemaan mahdollisimman tehokkaita ratkaisuja jo suunnittelun alkuvaiheessa. Suunnitteluratkaisujen optimointia tutkitaan erityisesti arkkitehdin ja rakennesuunnittelijan rajapinnassa tapahtuvan suunnittelun osalta.

Nykyään arkkitehdit voivat suunnitella hyvin monimuotoisia rakenteita tietokoneavusteisesti ilman, että niiden rakenteellista toimivuutta on varmennettu. Suunnitteluprosessin alkuvaiheessa tulee ottaa huomioon myös monia muita tekijöitä, kuten esimerkiksi rakennuksen esteettisyys ja energiatehokkuus. Arkkitehti aloittaa rakennuksen alustavan suun-

nittelun itsenäisesti ja välittää sitten tiedon suunnitelluista rakenteista rakennesuunnittelijalle. Tämän jälkeen rakennesuunnittelija voi aloittaa rakenteiden alustavan analysoimisen ja toteuttamiskelpoisuuden tutkimisen. Koska arkkitehdit ja rakennesuunnittelijat käyttävät suunnittelussa eri ohjelmia, on tämän suunnitteluvaiheen suorittaminen nykyisillä menetelmillä hyvin työlästä. Teknisten puutteiden vuoksi arkkitehdin alustavia suunnitelmia ei nykyisessä suunnitteluprosessissa optimoida rakennesuunnittelijan toimesta tyypillisesti ollenkaan. Algoritmiavusteisten suunnittelumenetelmien avulla voidaan kuitenkin hyödyntää MOO:n periaatteita. MOO mahdollistaa moniin erilaisiin tavoitteisiin pyrkivien suunnitteluratkaisujen vertailun, tiettyjen suunnittelutavoitteiden painottamisen ja vaihtoehtojen rakenteellisen toimivuuden nopeamman ja tarkemman tutkimisen. (Brown et al. 2016, s. 1103)

Tavallisesti rakennesuunnittelija saa suunnittelunsa lähtötiedoksi arkkitehdilta mallin, jonka rajoissa rakennesuunnittelijan on toimittava. Näiden rajojen mukaan rakennetaan suunnittelussa hyödynnettävät algoritmit, joiden avulla pyritään saavuttamaan rakennesuunnittelun kannalta oleelliset tavoitteet. Algoritmiavusteisessa suunnittelussa ongelmanratkaisualgoritmit voidaan jakaa kahteen pääluokkaan: heuristisiin ja metaheuristisiin algoritmeihin. Heuristiset algoritmit sisältävät tyypillisesti kokemusperäistä tietoa ja siksi ne harvoin antavat tulokseksi täysin optimaalista ratkaisua. Heuristiset algoritmit voidaan jakaa kolmeen kategoriaan:

1. Heuristinen algoritmi ilman ulkopuolista analyysiä.
2. Heuristinen algoritmi, joka käyttää kiinteää ulkopuolista analyysiä.
3. Heuristinen algoritmi, joka käyttää iteratiivista ulkopuolista analyysiä.

Algoritmit, jotka eivät hyödynnä ulkopuolista analyysiä (1), soveltuvat erityisesti ongelmiin, joita ei ole mahdollista ratkaista analyyttisesti. Suunnittelijan kokemus ja tieto syötetään itse algoritmiin eikä ulkopuolista analyysiä käytetä. Kun ratkaisu on jokseenkin odotettavissa, mutta ratkaisujoukko on suuri, on järkevää hyödyntää tietokoneen laskentatehoa ja numeerista ratkaisutapaa. Tätä lähestymistapaa käytetään esimerkiksi optimaalisen rakenteellisen muodon haussa. (Harding 2014, s. 20–21)

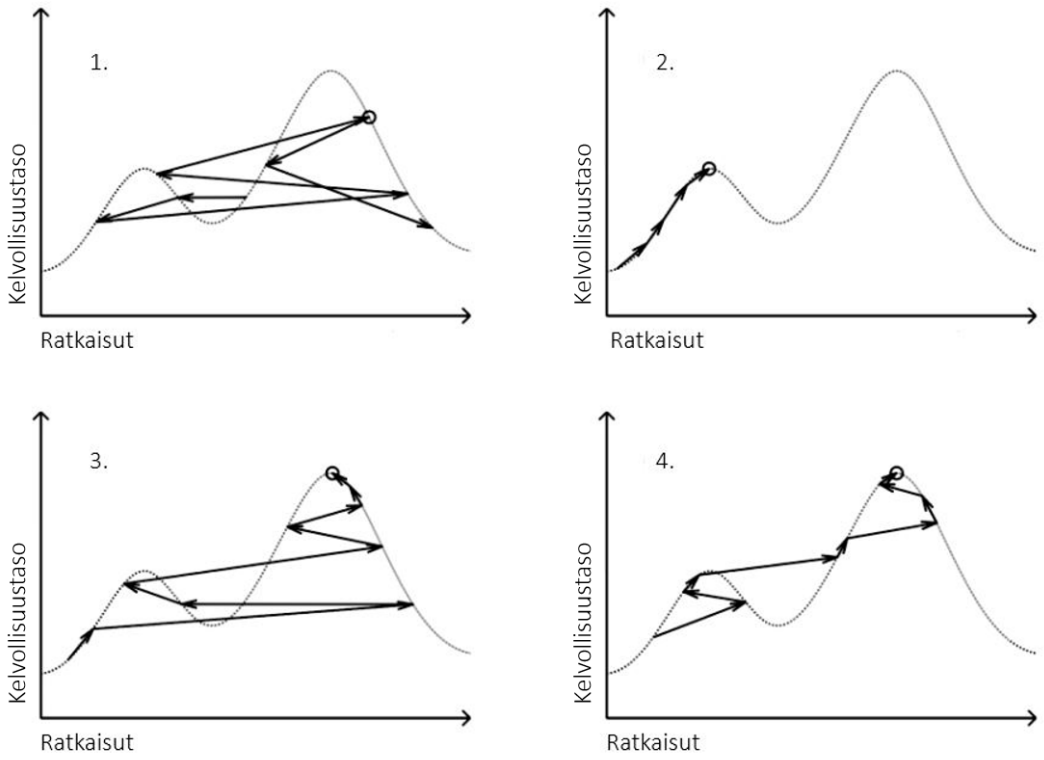
Algoritmiavusteisessa optimoinnissa voidaan hyödyntää myös ulkopuolista analyysiä. Tämä tarkoittaa sitä, että algoritmi hyödyntää jotain ulkopuolista tietoa, jota suunnittelija ei syötä suoraan algoritmiin. Tämä ulkopuolinen informaatio voi olla joko pysyvää tai muuttuvaa. Kun algoritmiin tuleva ulkopuolinen tieto pysyy vakiona, myös optimointi pysyy yksinkertaisempana. Kun algoritmien käyttämä data muuttuu optimoinnin edetessä, aiheuttaa se prosessiin jatkuvan palautekierteen. Tällöin prosessin etenemissuunnan ennustettavuus vähenee huomattavasti ja lopulliset tulokset voivat olla hyvin odottamattomia. Yleisesti ottaen heuristiset menetelmät soveltuvat hyvin rakennesuunnittelussa kohdattavien ongelmien ratkaisuun, koska ongelman kuvaaminen on usein melko helppoa ja tavat päästä optimaaliseen ratkaisuun ovat kohtalaisen hyvin tiedossa. (Harding 2014)

Metaheuristiset algoritmit eroavat heuristisista ennen kaikkea siinä, että ne irrottautuvat täysin siitä, miten ongelma ratkaistaan. Ne eivät tee mitään oletuksia ongelmasta tai sen mahdollisista ratkaisuista ja tarvitsevat näin toimiakseen suuren määrän mahdollisia ratkaisuja. Metaheuristiset menetelmät soveltuvat erityisesti tapauksiin, joissa mahdollinen ratkaisujoukko tunnetaan huonosti. Ne ovat helpommin muokattavissa kuin heuristiset algoritmit ja näin ollen soveltuvat hyvin yhteistyössä tapahtuvaan suunnitteluun, jossa sidosryhmien tietämys sijaitsee algoritmien ulkopuolella. Rakennesuunnittelussa algoritmien suorituskyvylle syötetään tyypillisesti kvantitatiiviset tavoitteet, joita etsitään asetettujen rajoitteiden ja parametrien rajaamasta otosavaruudesta. Metaheuristinen algoritmi suorittaa jonkinlaisen analyysin optimoinnin jokaisella iteraatiokierroksella, jotta se ymmärtää optimoinnin kelvollisuustason. Tätä kelvollisuustasoa kuvataan yleisesti fitness-funktion avulla ja se voi yksinkertaisimmillaan yksittäinen luku, jota kohti optimoinnissa pyritään. (Harding 2014, s. 36)

Metaheuristiset menetelmät käyvät käytännössä aina läpi koko ratkaisuavaruuden, mutta ne voivat nopeuttaa sitä erilaisten älykkäiden hakukeinojen avulla. Tarkastelemalla ratkaisuvaihtoehtoja älykkäästi voidaan saavuttaa hyvä lopputulos kohtuullisessa ajassa. Metaheuristisen menetelmän ratkaisu on usein valinta läpikäytävän ratkaisujoukon koon ja laskenta-ajan välillä. Tavallisesti mitä suurempi tarkasteltava ratkaisujoukko on, sitä tarkempi on myös ratkaisu. Pitkän laskenta-ajan vuoksi ratkaisuavaruuden täydellinen läpikäyminen on kuitenkin harvoin tarkoituksenmukaista. Neljä yleistä metaheuristista menetelmää ovat:

1. Väsytysmenetelmä (engl. brute-force).
2. Huipunhaku (engl. hill climbing).
3. Simuloitu jäähdytys (engl. simulated annealing).
4. Geneettiset algoritmit (engl. genetic algorithms).

Väsytysmenetelmä soveltuu erityisesti yksinkertaisten tehtävien ratkaisuun, koska siinä ratkaisujoukko käydään järjestelmällisesti läpi ilman erityistä sisäistä palauteprosessia. Väsytysmenetelmä on nimensä mukaisesti perusteellinen tapa löytää optimaalinen ratkaisu, mutta se vaatii myös pitkän laskenta-ajan. Laskenta-aikaa on mahdollista lyhentää niin sanottujen ahneiden algoritmien avulla. Esimerkiksi huipunhaku on tällainen menetelmä. Siinä algoritmi käy järjestelmällisesti läpi ratkaisujoukkoa tiettyyn suuntaan, kunnes tulokset alkavat huonota. Tämän menetelmän heikkoutena on se, että algoritmi ei hyväksy ollenkaan huononevia tuloksia vaan lopettaa toimintansa saavuttaessaan optimaalisen tuloksen. Tämä optimi voi kuitenkin olla vain lokaali huippu, jolloin voidaan jäädä hyvinkin kauas todellisesta optimista. (Harding 2014, 36–38) Edellä lueteltujen metaheuristisen menetelmien toimintaperiaatteet on esitetty kuvassa 26.

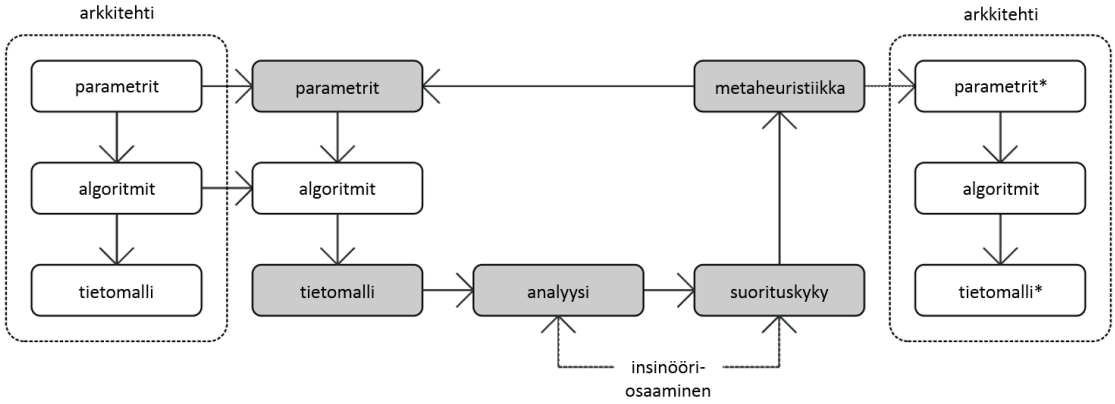


Kuva 26. *Erilaisten metaheurististen optimointialgoritmien toimintaperiaatteet perustuen lähteeseen (Harding 2014, s. 39).*

Yleisesti käytettyjä menetelmiä, jotka sijaitsevat pitkän laskenta-ajan omaavan väsytysmenetelmän sekä usein paikalliseen optimiarvoon päätyvien ahneiden algoritmien välillä ovat simuloitu jäähdytys ja evoluutioalgoritmit. Simuloidussa jäähdytyksessä algoritmi käy läpi ratkaisujoukkoa satunnaisesti, mutta pienentäen jatkuvasti askelkokoa ja lähentyen näin lokaalia optimia. Kun parametrit on asetettu ongelman kannalta oikeiksi, osaa algoritmi välttää paikallisen optimin ja jatkaa kohti todellista optimiarvoa. Myös evoluutioalgoritmit ja niihin kuuluvat geneettiset algoritmit pystyvät pääasiassa välttämään paikalliset optimiarvot. Evoluutioalgoritmit hyödyntävät samoja periaatteita kuin biologiassa tapahtuva evoluutio. Geneettinen algoritmi jakaa ratkaisujoukon sukupolviin, joissa sijaitsee aina tarkasteltava otospopulaatio. Sukupolvet kehittyvät ajan kuluessa aivan kuten biologiassa eli geenien vaihdon ja mutaatioiden kautta. Täten optimointi etenee kohti kehittyneintä tapausta eli optimiarvoa. (Harding 2014, s. 38–40)

Visuaaliset ohjelmointialustat sisältävät nykyään useita erilaisia metaheuristisia menetelmiä hyödyntäviä työkaluja. Esimerkiksi Grasshopperiin on saatavilla evoluutioalgoritmeja hyödyntävä työkalu nimeltään Galapagos, jonka avulla voidaan suorittaa erilaisia optimointitehtäviä. Suunnitteluprosessi etenee tyypillisesti siten, että arkkitehti luo ensin parametrisen mallin, jonka jälkeen rakennesuunnittelija suorittaa rakenteiden rakenteellisen optimoinnin. Optimointi tapahtuu liittämällä halutut parametrit metaheuristiseen algoritmiin ja asettamalla algoritmille tavoiteltava kelvollisuustaso eli niin sanottu fitness-

funktio. Fitness-funktio voi olla yksinkertaisimmillaan esimerkiksi tarkasteltavien rakenteiden paino. Jotta algoritmimalli saa tarvittavan palautteen rakenteiden suorituskyvystä, tulee siihen integroida jonkinlainen analyysityökalu tai linkittää jokin ulkopuolinen laskentaohjelma. (Harding 2014) Periaate arkkitehdin parametrisen mallin suorituskyvyn optimoinnista on esitetty kuvassa 27.



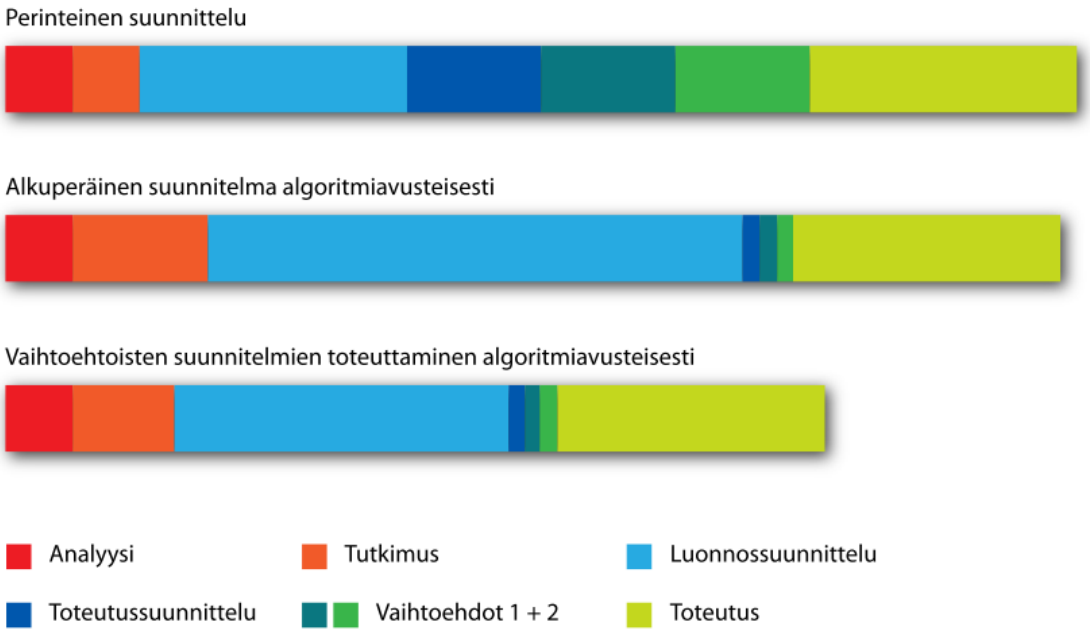
Kuva 27. *Arkkitehdin parametrisen mallin optimointiperiaate perustuen lähteeseen (Harding 2014, s. 45).*

Rakennesuunnittelun näkökulmasta rakenteiden optimointi keskittyy tyypillisesti niiden kustannusten minimointiin. Tällöin rakenteiden optimaalisuutta mitataan ainoastaan karkealla tasolla rakenneosien määrän ja painon mukaan. Kuten Harding (2014) työssään esittää, täysin optimaalisen tuloksen saavuttamiseksi rakennesuunnittelijan tulee osallistua suunnitteluun jo arkkitehtisuunnittelun alkuvaiheessa. Jos arkkitehdin alustavat suunnitelmat lukitaan ennen kuin rakennesuunnittelija liittyy mukaan suunnitteluun, on täysin optimaalisen tuloksen saavuttaminen käytännössä mahdotonta. Rakennesuunnittelija voi optimoida arkkitehdin valmiiksi suunnittelemia rakenteita, mutta siinä saavutettavat hyödyt ovat suhteellisen pieniä kokonaisuuteen verrattuna. Harding käy työssään läpi projekteja, joissa hyödynnettiin algoritmiavusteista suunnittelua rakenteiden optimaaliseen suunnitteluun heti suunnittelun aloituksesta. Suunnitteluprosessin monimutkaisuuden vähentämiseksi insinöörit vastasivat kuitenkin suunnittelusta yksin eikä projekteihin osallistunut ollenkaan arkkitehteja.

3.4 Algoritmiavusteisen suunnittelun hyödyt ja haasteet

Algoritmiavusteinen suunnittelu sisältää monia hyötyjä ja kaikkea sen sisältämää potentiaalia ei ole vielä edes päästy kokeilemaan. Yksi suurimmista algoritmisen suunnittelun vahvuuksista on mallin muokkaamisen helppous. Algoritmit saavat tyypillisesti lähtötiedokseen parametreja, jotka ovat helposti muokattavissa koko prosessin ajan. Algoritmit ja niiden ohjaamat mallit päivittyvät automaattisesti parametreja muutettaessa, jolloin suunnittelija näkee jatkuvasti tekemiensä muutosten vaikutukset. Tämä mahdollistaa myös useiden erilaisten vaihtoehtojen nopean vertailun ja nopeuttaa näin iteratiivista

suunnitteluprosessia huomattavasti. (Tanska & Österlund 2014) Algoritmiavusteisen suunnitteluprosessin ja sen osatekijöiden suhteet on esitetty kuvassa 28. Kuvassa näkyvät myös perinteisen suunnitteluprosessin sekä valmiita työkaluja hyödyntävän algoritmiavusteisen suunnitteluprosessin kestot.



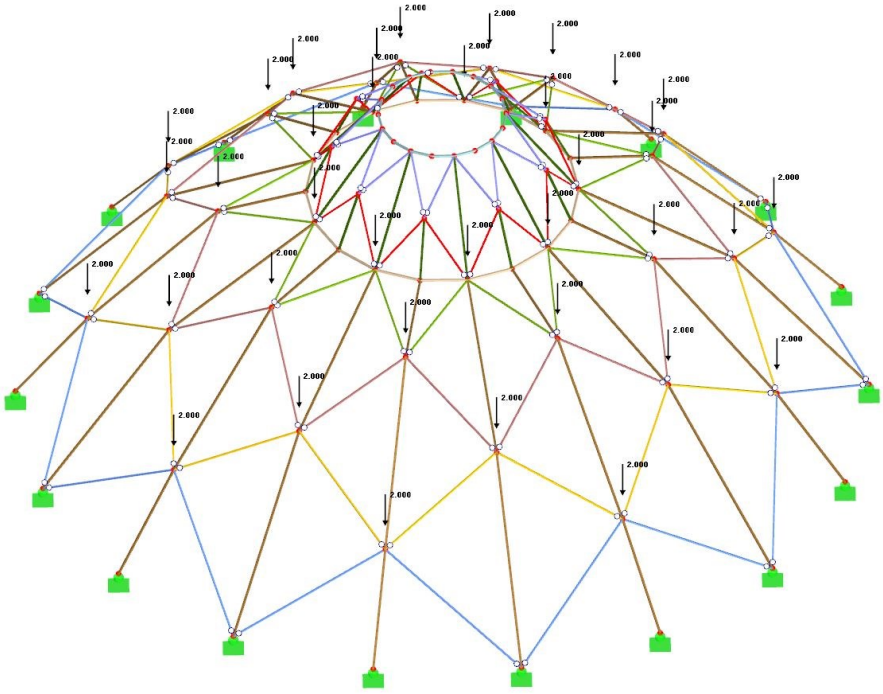
Kuva 28. *Erilaisiin suunnitteluprosesseihin ja niiden vaiheisiin kuluvan ajan vertailu (Tanska & Österlund, s. 24).*

Toinen algoritmipohjaisen suunnittelun vahvuus on suunnittelutiedon yksinkertaisuus. Visuaalista ohjelmointia hyödynnettäessä suunnittelutieto on hyvin yksinkertaisessa muodossa ja siten myös helposti muokattavissa. Yksinkertaiset algoritmimallit ovat tiedostokooltaan hyvin pieniä, mutta sisältävät silti kaiken tarvittavan datan. Algoritmit ovatkin tiedostokooltaan vain murto-osan perinteisiin BIM-malleihin verrattuna. Ne sisältävät ainoastaan säännöt ja ohjeet, joiden avulla perinteiset tieto- ja laskentamallit luodaan. Tätä hyvin yleisessä muodossa olevaa tietoa voidaan helposti hyödyntää myös muissakin projekteissa, toisin kuin perinteisiä BIM-malleja. Kun tiedostokoot pysyvät pieninä, tiedonsiirto ja -käsittely on nopeaa. Tämä mahdollistaa esimerkiksi vaivattoman tiedonsiirron eri suunnitteluosapuolien ja -mallien välillä. Suunnittelutieto voidaan jakaa useisiin algoritmimalleihin esimerkiksi rakennekokonaisuuden mukaan. Tällöin myös niiden generoivat tieto- ja laskentamallit säilyvät kevyempinä ja täten myös joustavampina käyttää. (Van der Heijden 2015)

Algoritmiavusteisen suunnittelun ominaisuuksista johtuen suunnittelussa voidaan helposti palata aikaisempiin suunnitteluvaiheisiin. Tämä onnistuu sekä algoritmimallien että yksittäisten visuaalisten ohjelmakomponenttien kohdalla. Rakennesuunnittelun iteratiivisesta luonteesta johtuen tämä luo suunnitteluun täysin uudenlaisen mahdollisuuden. Perinteisessä tietomallintamisessa iteratiivinen prosessi eli yrityksen ja erehdyksen kautta

tapahtuva suunnittelu koostuu tyypillisesti tehtävistä ja lähtötilanteeseen palauttavista vastatehtävistä. Jos esimerkiksi suunnittelija siirtää jonkin rakenneosan uuteen sijaintiin ja haluaa myöhemmin palauttaa sen takaisin alkuperäiseen paikkaan, joutuu hän tekemään kaksi siirtotehtävää. Visuaalista ohjelmointia hyödynnettäessä suunnittelija kuitenkin selviää edellä esitetystä esimerkistä ainoastaan yhdellä komennolla. Algoritmissa säilyy kaikki mallin luomiseen ja muokkaamisen käytetyt komponentit ja niiden vastaanottamat syötteet ja lähettämät tulosteet. Näin ollen pienienkin muokkausten jälkeisiin vaiheisiin on hyvin helppoa ja nopeaa palata.

Algoritmeihin pohjautuva suunnittelu mahdollistaa myös haastavien muotojen kuvaamisen. Erityisesti kaarevien muotojen luominen perinteisissä tietomallinnusohjelmissa on usein lähes mahdotonta. Nykyään esimerkiksi rakennesuunnittelussa paljon käytetty Tekla Structures tukee myös kaarevia muotoja, mutta niiden luominen ohjelman omassa käyttöliittymässä on vaikeaa, ellei jopa mahdotonta. Kaarevat muodot voidaankin helposti generoida algoritmien avulla ja ohjata Tekla-objektit seuraamaan näitä muotoja. Algoritmien avulla vaativamman geometrian luominen helpottuu myös laskentamallien osalta. Koska FEM-ohjelmien mallinnustyökalut ovat usein hyvin yksinkertaisia, on vaikeiden geometrioiden luominen työlästä. Näin ollen algoritmit tuovat merkittävän hyödyn myös laskentamallien luomiseen. Optimaalisessa tilanteessa samojen algoritmien avulla voidaan ohjata samanaikaisesti sekä tieto- että laskentamallia. Esimerkki algoritmiavusteisesti tuotetusta, haastavan geometrian omaavasta FEM-mallista on esitetty kuvassa 29.



Kuva 29. *Algoritmiavusteisesti tuotettu FEM-malli.*

Vaikka algoritmiavusteinen suunnittelu sisältää paljon hyötyjä ja mahdollisuuksia, on sen käytössä myös tiettyjä haasteita. Koska algoritminen suunnittelu on vielä hyvin uusi asia rakennusalalla, varsinkin rakennesuunnittelijoiden keskuudessa, ei siihen ole vielä ehtinyt muodostua vakiintuneita tapoja toimia. Haasteita aiheuttavat sekä yksittäisten suunnittelutehtävien suorittaminen että koko suunnitteluprosessin läpikäyminen. Ohjelmointia hyödyntävä algoritminen suunnittelu on hyvin erilaista perinteiseen suunnitteluun verrattuna. Se tarkoittaa sitä, että suunnittelijat joutuvat opettelemaan täysin uudenlaisen tavan tehdä suunnittelua. Toisaalta visuaalinen ohjelmointi on huomattavasti helpommin omaksuttavissa kuin tekstimuotoinen ohjelmointi. Suunnittelijalla ei siis tarvitse olla minkäänlaista ohjelmointitaustaa algoritmiavusteisen suunnittelun syvälliseksi oppimiseksi. Myös algoritmiavusteisen suunnitteluprosessin hyödyntämisestä ja integroimisesta muuhun suunnitteluun, on hyvin vähän kokemusta. Haasteita aiheuttavat esimerkiksi suunnittelurajapintojen määrittäminen eri osapuolien ja ohjelmien välillä. Algoritmipohjaista suunnittelutietoa käytettäessä ja jaettaessa tulee myös ottaa uudella tavalla huomioon suunnitteluosapuolien tietosuoja ja kehitettyjen algoritmien tekijänoikeudet.

Visuaalinen ohjelmointi mahdollistaa suunnitteluratkaisujen vertailun lähtötietona toimivia parametreja muuttamalla. Haasteena on kuitenkin määritellä vapaat parametrit niin, että malli pystytään muokkaamaan kaikkiin haluttuihin muotoihin. Kun algoritmien muodostaman suunnittelukokonaisuuden muokkaaminen ei onnistu olemassa olevien parametrien avulla, joudutaan usein muokkaamaan koko algoritmirakennetta. Jos algoritmimalli on rakenteellisesti monimutkainen tai vain visuaaliselta ilmeeltään sekava, on malliin tehtävien muutosten tekeminen haastavaa. Erityisesti kun algoritmeja tarkastelee joku muu kuin niiden alkuperäinen luoja, saattaa muutosten tekeminen muodostua suureksi ongelmaksi. Tällöin saattaakin olla jopa järkevintä muodostaa täysin uudet algoritmit alusta. Tämä tarkoittaa kuitenkin sitä, että algoritmiavusteisen suunnittelun avulla haettavat hyödyt menetetään. (Davis et al. 2011, 363)

Jotta algoritmit ja niiden lähtötietona toimivat parametrit toimisivat riittävän joustavasti koko suunnitteluprosessin ajan, tulee suunnittelun etenemistä ja mahdollisia muutoksia ennakoida prosessin alusta alkaen. Tämä on kuitenkin haasteellista erityisesti suurempien rakennekokonaisuuksien kohdalla. Suunnittelun ennustettavuutta vähentää myös useiden eri osapuolien osallistumien suunnitteluun. Esimerkiksi rakennesuunnittelu tapahtuu lähes aina arkkitehdin lähtötietojen mukaan ja näin ollen mahdolliset suunnittelumuutokset voivat olla myös täysin itsestä riippumattomia. Käytännössä koko suunnitteluprosessin ennustaminen onkin mahdotonta ja siitä syystä myös algoritmeihin joudutaan usein tekemään suuriakin muutoksia suunnittelun edetessä. (Davis 2013)

4. CASE: UIMAHALLIN ALGORITMIAVUSTEINEN SUUNNITTELU

Algoritmiavusteista suunnittelua voidaan hyödyntää monenlaisten rakenteiden kohdalla, mutta sen voi katsoa soveltuvan erityisesti vaativien muotojen sekä toistuvien rakenteiden suunnitteluun. Tämä johtuu pääasiassa siitä, että algoritmipohjaisten menetelmien avulla kyseisten rakenteiden suunnittelu on huomattavasti helpompaa perinteisiin menetelmiin verrattuna. Esimerkiksi hyvin haastavien muotojen kohdalla tieto- ja laskentamallin rakentaminen voi olla lähes mahdotonta perinteisin menetelmin. Algoritmiavusteisen suunnittelun parametrinen luonne tuo myös suurta helpotusta toistuvien rakenteiden suunnitteluun. Suunniteltavien rakenteiden muodosta ja ominaisuuksista riippumatta algoritmiset menetelmät voivat tuoda merkittäviä etuja koko suunnitteluprosessin etenemiseen. Tutkimuksessa ei pyritä suunnittelemaan valmiita, täysin toteutuskelpoisia rakenteita vaan keskitytään ennen kaikkea tarkastelemaan suunnitteluprosessin eri vaiheita ja testaamaan algoritmisen suunnittelun soveltuvuutta niihin.

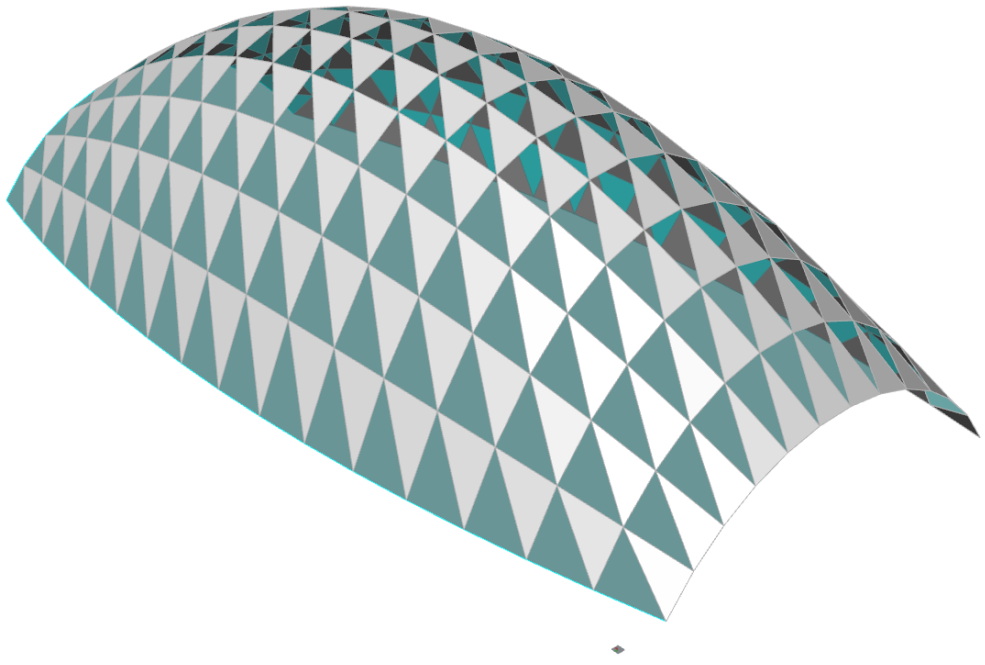
Tässä työssä suunnitteluprosessia tarkastellaan fiktiivisen suunnittelukohteen avulla. Tarkoituksena on testata ja analysoida algoritmisten menetelmien soveltuvuutta rakennesuunnittelun eri vaiheisiin aina arkkitehdin lähtötiedoista liitosten toteutussuunnitteluun. Suunnitteluprosessiin osallistuu sekä arkkitehti että rakennesuunnittelija. Arkkitehti hyödyntää suunnittelussaan myös algoritmisia menetelmiä ja tarkoituksena on testata niiden tehokasta hyödyntämistä rakennesuunnittelun lähtötietona. Tutkimuksen tavoitteena onkin rakennesuunnitteluprosessin lisäksi tarkastella älykästä vuorovaikutusta arkkitehdin ja rakennesuunnittelijan välillä. Suunnitteluosapuolien välisessä tiedonsiirrossa hyödynnetään uutta pilvipohjaista tiedonjakotapaa, joka voidaan myös linkittää älykkäästi algoritmiseen suunnitteluprosessiin.

Tässä luvussa esitetään työssä toteutettu tapaustutkimus. Luvussa esitellään tutkimuksen toteutustapa ja käydään läpi sen merkittävät ominaisuudet ja vaiheet yksi kerrallaan. Aluksi kuvataan tutkimuksessa hyödynnettävä suunnittelukohde, johon kaikki työssä toteutettava suunnittelu kohdistuu. Suunnittelukohde toimii viitekehyksenä suunnitteluprosessin etenemistä tarkastellessa sekä kehitettäviä työkaluja testatessa. Seuraavaksi käsitellään tutkimuksessa läpi käytävän suunnitteluprosessin ominaisuuksia. Käsittely kohdistetaan tutkimuksessa käytettäviin ohjelmiin ja lisäosiin sekä tiedonsiirtotapoihin. Tämän jälkeen tarkastellaan teräsrakenteisen kattorakenteen mitoitusta ja mallinnusta. Lopuksi esitetään vielä tutkimuksessa toteutettava liitoksen suunnitteluprosessi. Liitoksen suunnittelussa edetään aina laskentamallista saatavista kuormituksista liitoksen kestävyystarkasteluun ja yhä liitoksen tietomallintamiseen.

4.1 Suunnittelukohteen esittely

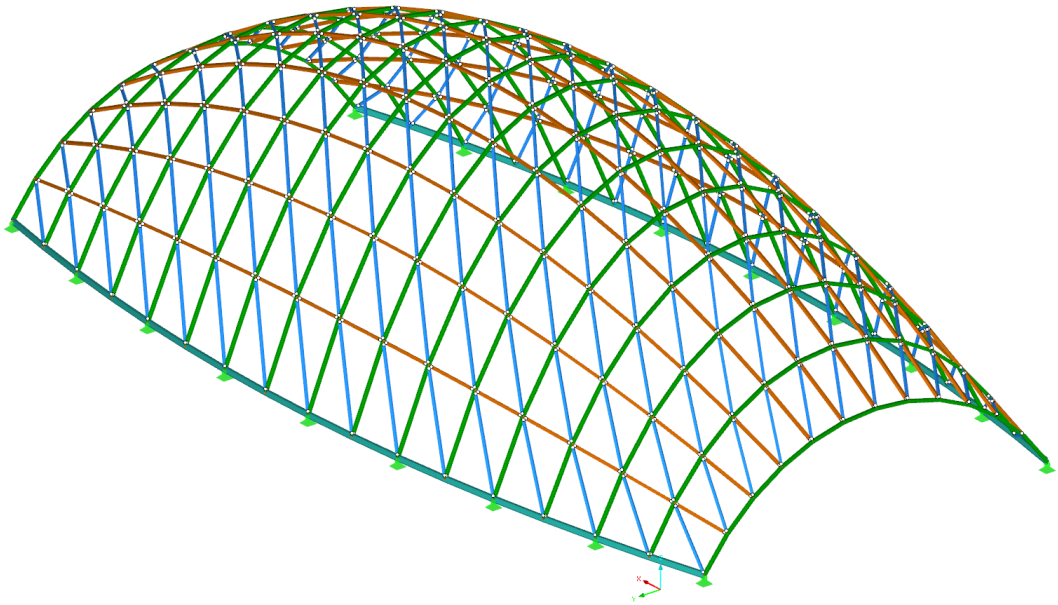
Tässä tutkimuksessa käydään läpi algoritmiavusteista suunnitteluprosessia kuvitteellisen suunnittelukohteen avulla. Suunnittelukohteeksi valikoitui uimahalli, jonka merkittävä ominaisuus on monimuotoinen teräsrakenteinen kattorakenne. Suunnittelun painopiste kohdistetaan erityisesti haastavan kattorakenteen suunnitteluun, jotta päästään näkemään algoritmiavusteisen suunnittelun todelliset hyödyt ja haasteet. Kattorakenne sisältää sekä vaativia muotoja että toistuvia rakenneosia, minkä johdosta sen koetaan olevan oivallinen algoritmiavusteisen suunnitteluprosessin testauskohde. Suunnittelussa ei pyritä täydelliseen, toteutettavaan rakenteeseen vaan tarkoituksena on testata kattorakenteen avulla tyypillisiä suunnittelutehtäviä. Suunniteltava kattorakenne on haastavan muotonsa johdosta oivallinen esimerkki algoritmisten menetelmien mahdollisuuksista. Perinteisiä suunnittelumenetelmiä käytettäessä pelkästään geometrian luominen ja mahdollinen muokkaaminen olisivat hyvin työläitä tehtäviä.

Kattorakenne koostuu kaarevasta pinnasta, joka jaetaan pitkittäis- ja poikittaissuuntaisiin osiin. Tämän jälkeen muodostuneiden ruutukuvioiden lävitse luodaan vielä vinosauvat, jolloin kattorakenne koostuu säännöllisistä kolmiopinnoista. Kattopinnan viivajako toteutetaan parametrisesti, minkä ansiosta viivageometrian tiheysjaon muokkaaminen on hyvin helppoa ja nopeaa. Arkkitehti luo algoritmiavusteisesti kaarevan kattopinnan ja generoi siihen edellä esitetyn viivageometrian. Tämän jälkeen rakennesuunnittelija voi hyödyntää tätä viivageometriaa oman suunnittelunsa lähtötietona. Pinnan tiheysjaon määrittävät kaksi numeroparametria säilyvät avoimina, mikä tarkoittaa sitä, että rakennesuunnittelija voi muokata tiheysjakoa oman mielensä mukaan. Kuvassa 30 esitetään arkkitehdin luoma kattorakenteen geometria.



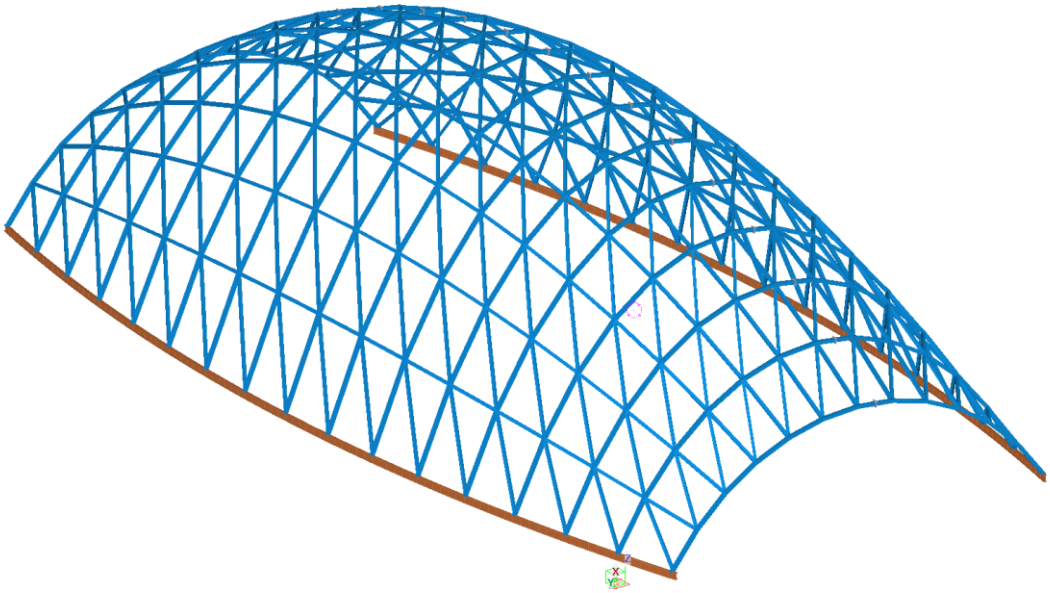
Kuva 30. *Kattorakenteen yksinkertainen geometriamalli.*

Kun viivageometria on muodostettu, voidaan sitä helposti hyödyntää myös laskenta mallin luomiseen. Arkkitehdin alkuperäinen viivageometria on muodostettu puumuotoon, jossa ensimmäisellä oksalla ovat vinosauvat, toisella poikittaissuuntaiset sauvat ja kolmannella pitkittäissuuntaiset sauvat. Jotta laskentamallista voidaan luoda rakenneteknisesti toimiva, ryhmitellään geometria neljään osaan. Ensimmäisellä ja toisella oksalla sijaitsevat viivat säilyvät muuttumattomina, mutta kolmas oksa jaetaan kahteen osaan. Kattorakenteen alimmista pitkittäissuuntaisista sauvoista eli niin sanotuista tukipalkeista muodostetaan oma ryhmä, jotta niille voidaan määrittää muista sauvoista poikkeavat ominaisuudet. Kuvassa 31 esitetään kattorakenteen viivageometriasta muodostettava laskentamalli. Rakennejärjestelmä koostuu IPE-profiililla toteutetuista tukipalkeista sekä neliönmuotoisilla putkiprofiileilla muodostetusta ristikkorakenteesta. Kattorakenne tukeutuu molemmilla puolilla yhdeksään tukipisteeseen.



Kuva 31. *Kattorakenteen laskentamalli.*

Tukipalkkien päälle tukeutuvat poikittaissuuntaiset kaaripalkit muodostuvat kahdesta kokoonpanosta. Kokoonpanot koostuvat suorista terässauvoista, jotka hitsataan toisiinsa kiinni. Teräskokoonpanot kiinnittyvät toisiinsa tukipalkkien puolivälissä jatkosliitoksen avulla. Kattorakennetta kannattelevat tukipalkit toteutetaan kaarevina rakenteina. Palkkien päitä jatketaan, jotta myös päätyjen kaaripalkit pystyvät tukeutumaan riittävästi niiden päälle. Rakenneosat sijoittuvat pääasiassa laskentamallin sauvojen kanssa samoihin sijainteihin, mutta joitakin pieniä muutoksia on mahdollista tehdä. Esimerkiksi vinosauvojen sijaintia voidaan muokata siten, että sauvat eivät törmää keskenään. Algoritmien generoima rakennemalli esitetään kuvassa 32. Rakennemallin muodostamiseen hyödynnetään tietomallinnuksessa yleisesti käytettävää Tekla Structuresia.



Kuva 32. *Kattorakenteen rakennemalli.*

Tässä tutkimuksessa toteutetaan useita erilaisia variaatioita edellä esitetystä teräsrakenteisesta kattorakenteesta. Katon muoto ja muut ominaisuudet eivät kuitenkaan ole tämän työn kannalta oleellisia, koska tarkoituksena on ennen kaikkea kokeilla ja kehittää algoritmiavusteisen suunnitteluprosessin toimintaa.

4.2 Suunnitteluprosessin ominaisuudet

Kuten luvussa 3 esitetään, algoritmiavusteinen suunnitteluprosessi koostuu tyypillisesti monista algoritmimalleista sekä niiden avulla ohjattavista perinteisistä suunnittelumalleista. Tieto linkittyy algoritmien välityksellä toisiinsa siten, että koko suunnitteluprosessista voidaan muodostaa automaattisesti päivittyvä systeemi. Kaiken perustana toimivat visuaalisen ohjelmointialustan avulla muodostettavat algoritmimallit sekä niiden rajapinnat muihin käytettäviin ohjelmiin. Algoritmisen suunnittelun kohdalla onkin tärkeää kiinnittää huomiota siihen, miten suunnittelutietoa jaotellaan ja ryhmitellään. Tämä pätee sekä yhden tiedoston sisällä tapahtuvaan ryhmittelyyn että käytettävien mallien väliseen jaotteluun.

Yksi olennainen osa algoritmiavusteista suunnittelua on algoritmisten mallien automaation aste. Kun suunnittelija luo algoritmiavusteisesti toimivaa suunnitteluprosessia, joutuu hän määrittelemään, mitkä ominaisuudet toimivat automatisoidusti ja mihin suunnittelija voi itse vaikuttaa. Tämä tarkoittaa sitä, että suunnitteluprosessista pyritään tekemään mahdollisimman tehokas ilman, että sen joustavuus kärsii liikaa. Koska suunniteltavat rakenteet ja niiden muodostamat kokonaisuudet eroavat aina jonkin verran toisistaan, on tämän rajan määrittäminen hyvin vaikeaa. Suunnittelussa kannattaakin pyrkiä automatisoimaan täysin ainoastaan pieniä kokonaisuuksia kerrallaan, jotta suunnitteluprosessi pysyy riittävän joustavana ja pystyy mukautumaan haluttuihin muutoksiin. Al-

goritmeja luotaessa on tärkeää huomioida kaikki vapaaksi jätettävät parametrit sekä niiden mahdolliset vaihteluvälit. Tällöin algoritmien avulla saavutettava rakenteiden parametrinen muokkaaminen pystytään toteuttamaan tarkoituksenmukaisesti kaikissa tilanteissa.

Kuten kuvassa 28 esitetään algoritmiavusteinen suunnittelu nopeuttaa huomattavasti vaihtoehtoisten suunnitelmien toteuttamista. Tämä voi tarkoittaa sekä yksittäisen projektin sisällä tapahtuvaa suunnitelmien muokkaamista että kokonaan toisen hankkeen puitteissa tapahtuvaa suunnittelua. Näin ollen algoritmeja luotaessa on hyvä pitää mielessä kaikki mahdolliset tapaukset, joihin niiden tulee pystyä reagoimaan. Tämä on usein mahdotonta ja siksi algoritmimalleja luotaessa on tärkeää kiinnittää huomiota mallien selkeyteen. Kun algoritmit nimetään, jaotellaan ja ryhmitellään selkeästi, voidaan niitä käyttää helpommin hyödyksi myös tulevaisuudessa. Myös koko prosessin selkeyteen tulee kiinnittää huomiota. Tämä tapahtuu pääasiassa jakamalla algoritmeja erillisiin algoritmimalleihin, joita voidaan käsitellä itsenäisinä kokonaisuuksina. Rakennesuunnittelun ominaisuuksista johtuen algoritmien avulla ohjataan tyypillisesti useita ulkopuolisia ohjelmistoja. Tällöin tulee kiinnittää huomiota myös ohjelmistojen ja tiedostojen väliseen tiedonsiirtoon.

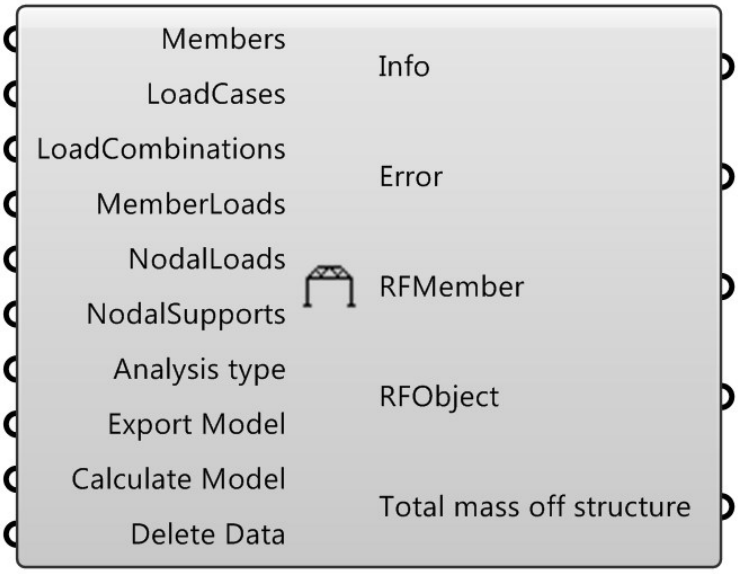
4.2.1 Käytettävät ohjelmat ja lisäosat

Tässä tutkimuksessa algoritmiavusteinen suunnittelu toteutetaan Rhinoceros 3D -ohjelman lisäosalla Grasshopper. Grasshopper sisältää lukuisia tiedon käsittelyyn ja muokkaamiseen käytettäviä komponentteja. Tässä tutkimuksessa käytetään pääasiassa vain valmiita ohjelmakomponentteja, jolloin suunnittelijalla ei tarvitse olla erityistä osaamista tekstimuotoisesta ohjelmoinnista. Komponenttivalikoimaa on mahdollista laajentaa myös useiden erilaisten lisäosien (engl. plug-in) avulla. Lisäosat toimivat usein esimerkiksi rajapintoina Grasshopperin ja muiden ohjelmistojen välillä. Lisäosa on tyypillisesti joukko Grasshopperin kankaalle sijoitettavia visuaalisia ohjelmakomponentteja, joiden toimintaperiaate ei juurikaan eroa alkuperäisistä komponenteista. Saatavilla olevien lisäosien sisältö ja laajuus vaihtelevat suuresti ja niitä ovatkin kehittäneet niin yksityishenkilöt kuin suuret ohjelmistoyrityksetkin. Pääasiassa lisäosat ovat ilmaisia ja ne ovat ladattavissa Food4Rhino-sivustolta. Tässä tutkimuksessa hyödynnetään monia Grasshopperin toimintaa laajentavia lisäosia ja niistä työn kannalta merkittävimmät ovat:

1. Grasshopper-RFEM Link.
2. Grasshopper-Tekla Live Link.
3. Macaw.
4. MetaHopper.
5. Speckle.

Grasshopper-RFEM Link (GH-RFEMLink) on A-Insinöörit Suunnittelu Oy:n kehittämä lisäosa, jonka avulla voidaan ohjata algoritmisesti Dlubal Software Inc:n RFEM-ohjel-

mistoa. RFEM on rakenteiden mitoitukseen ja analysointiin erikoistunut FEM-laskenta-ohjelma, joka soveltuu useiden erityyppisten rakenteiden suunnitteluun. GH-RFEMLink toimii rajapintana Grasshopperin ja RFEM:n välillä ja sen avulla on tällä hetkellä mahdollista rakentaa erilaisia laskentamalleja sekä mitoittaa sauvamaisia teräsrakenteita. GH-RFEMLink hyödyntää RFEM:n RF-COM -lisäosia, joka on sen avoin ohjelmoitava rajapinta. Lisäosan avulla on mahdollista luoda Grasshopperissa FEM-mallin sisältämät sauvat, niihin kohdistuvat piste- ja viivakuormat sekä rakenneosien ulkoiset tuennat. Lisäosan avulla voidaan määrittää myös muut laskentamallin kannalta olennaiset ominaisuudet, kuten esimerkiksi kuormaluokat ja kuormitusyhdistelyt. Laskentamallin luomiseen käytettävä komponentti on esitetty kuvassa 33.



Kuva 33. *RFEM-mallin luomiseen käytettävä komponentti.*

Kuten kuvasta 33 nähdään, voidaan lisäosan avulla käynnistää FEM-mallin laskenta. Kun laskenta on suoritettu, voidaan linkin välityksellä siirtää rakenteisiin kohdistuvat kuormitukset takaisin Grasshopperiin. Tämän lisäksi RFEM-linkin avulla voidaan ohjata RFEM:n RF-STEEL EC3 -lisäosaa ja siten suorittaa teräsosien Eurokoodin (EN 1993-1) mukainen mitoitus.

Grasshopper-Tekla Live Link on Trimblen kehittämä lisäosa Grasshopperin ja Tekla Structuresin välille. Linkki mahdollistaa tietomallin luomisen Teklaan Grasshopperia hyödyntämällä. Linkki sisältää lähes kaikki tietomallin luomiseen ja muokkaamiseen tarvittavat komponentit. Tällä hetkellä lisäosan ainoat rajoitteet ovat pultti- ja hitsauskomponenttien puuttuminen, mutta niidenkin luominen onnistuu osana liitoskomponentteja. Grasshopper-Tekla Live Linkin avulla on mahdollista esimerkiksi luoda erilaiset teräs-, betoni- ja puuosat sekä määrittää niiden attribuuttitiedot. Linkki mahdollistaa Grid-verkkojen, erilaisten liitoskomponenttien sekä betoniraudoitteiden luomisen. Lisäksi lisäosan avulla voidaan linkittää Teklassa luotuja osia ja komponentteja Grasshopperiin ja muuntaa Tekla-objekteja Grasshopperin ymmärtämäksi yksinkertaiseksi geometriatiedoksi.

(Grasshopper-Tekla Live Link) Grasshopper-Tekla Live Linkin komponenttivalikko on esitetty kuvassa 34.

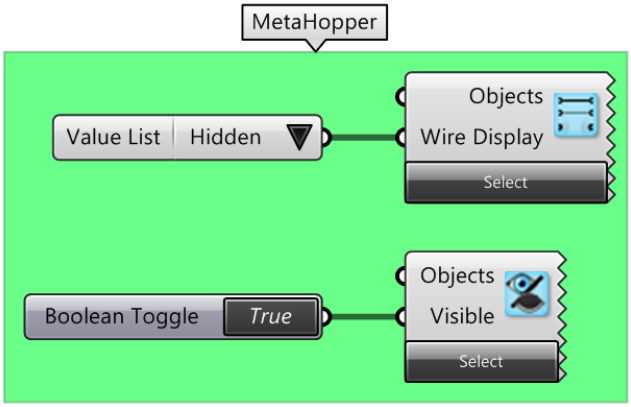
Kuva 34. *Grasshopper-Tekla Live Link komponenttivalikko.*

Kolmas tässä työssä eniten käytetyistä Grasshopperin lisäosista on Macaw, joka toimii rajapintana Grasshopperin ja Mathcad-laskentaohjelman välillä. Mathcad on rakennesuunnittelussa paljon käytetty tekninen laskentaohjelma, joka mahdollistaa matemaattisten ja tieteellisten laskentakaavojen käytön ja laskennan selkeän visuaalisen esittämisen. Mathcad tunnistaa myös käytetyt yksiköt ja osaa muuntaa ne oikeaan muotoon erityyppisissä laskelmissa. Rakennesuunnittelussa käytetään hyvin paljon erilaisia Mathcad-pohjia niiden helppokäyttöisyyden ja selkeän esitystavan vuoksi. Mathcadin avulla luotavat laskentatiedostot toimivat oivallisesti myös laskennan raportoinnissa. Macaw-lisäosa voidaan linkittää ainoastaan Mathcad Prime -ohjelmaan. Esimerkiksi vanhempaan ja paljon käytettyyn sovellusversioon Mathcad 15:ta ei löydy ohjelmistotukea. Toisaalta myös Mathcad 15 -versiolla luotuja laskentapohjia voidaan hyödyntää muuntamalla ne ensin Primen käyttämään tiedostomuotoon. (Macaw)

Macaw-lisäosan komponentit voidaan jakaa karkeasti kahteen osaan: Mathcadia ohjaaviin syötteisiin sekä sieltä tulosdataa palauttaviin tulosteisiin. Yksinkertaisimmillaan Mathcad-tiedostosta valitaan halutut syötteet ja niille annetaan haluttu lukuarvo ja yksikkö niihin liitetyn tunnuksen avulla. Tämän jälkeen lisäosan avulla voidaan suorittaa laskenta, joka Mathcadin tapauksessa tapahtuu hyvin nopeasti. Lopuksi laskentapohjasta voidaan hakea ennalta määritetyt tulosteet takaisin Grasshopperiin. Myös tulosteista saadaan niin tunnus, lukuarvo kuin yksikkökin. Mathcadissa suoritettavat laskutehtävät ovat usein melko yksinkertaisia ja ne voitaisiin hyvin suorittaa myös suoraan Grasshopperissa ilman ylimääräistä rajapintaa. Mathcad toimii kuitenkin käytännöllisenä raportointialustana ja siihen on saatavilla lukuisia valmiita laskentapohjia, joita voidaan hyödyntää sellaisenaan. Yksinkertainen esimerkki Mathcadin syöte, laskenta ja tuloste komponenteista on esitetty kuvassa 35.

Kuva 35. *Esimerkki Macaw-lisäosan toiminnasta.*

Listalla neljäntenä oleva MetaHopper toimii ennen kaikkea algoritmimallin luomista ja hallinnointia helpottavana työkaluna. Se sisältää joukon komponentteja, joiden avulla voidaan muokata Grasshopper-komponentteja dynaamisesti. Lisäosan avulla voidaan valita käytettävistä komponenteista halutut esimerkiksi niiden tyypin tai ryhmän mukaan ja muokata niiden ominaisuudet omien tavoitteiden mukaisiksi. MetaHopperin komponenttien avulla voidaan esimerkiksi muokata dynaamisesti ryhmien värejä, geometrian esikatselua, linkkien näkymätapaa sekä komponenttien ja niiden osien nimiä. (MetaHopper) Näiden asioiden selkeä määrittely lisää huomattavasti algoritmimallin ymmärrettävyyttä ja helpottaa tarkastelua, kuten kappaleessa 3.2.3 kuvataan. Kuvassa 36 esitetään esimerkki siitä, miten MetaHopperin komponenteilla voidaan muokata valittujen komponenttien ja niiden välisten linkkien näkymätilaa. Myös komponenttien muodostaman ryhmän väri määräytyy dynaamisesti ryhmälle annetun nimen mukaan.



Kuva 36. *Esimerkki MetaHopperin avulla tapahtuvasta näkymäasetusten muokkaamisesta.*

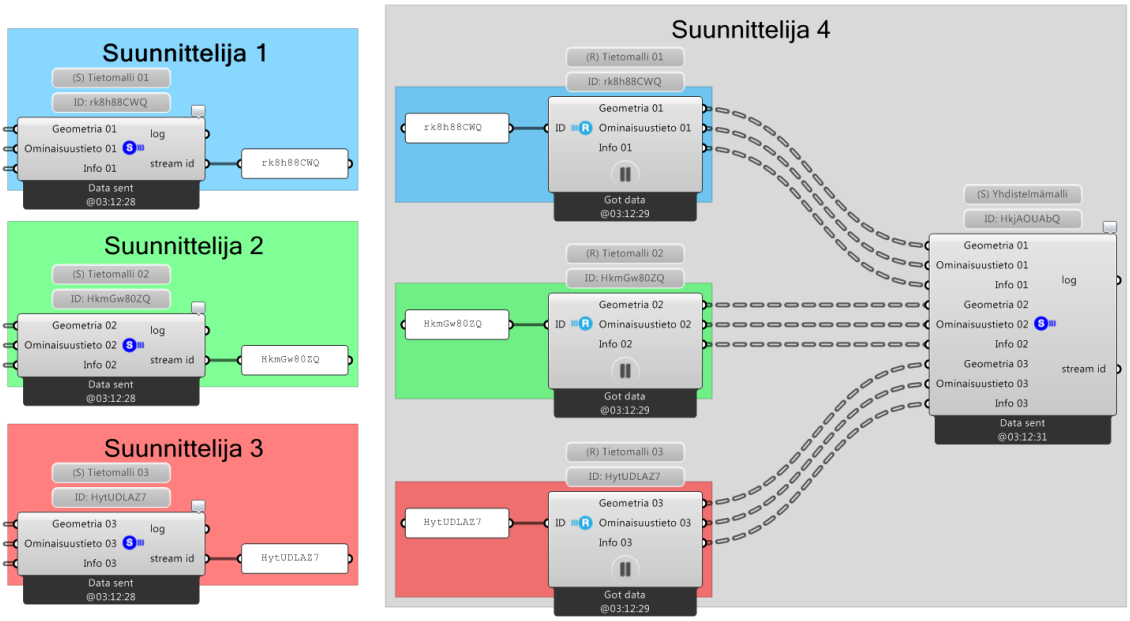
Listan viimeinen lisäosa on algoritmimallien pilvipohjaiseen jakamiseen tarkoitettu Speckle. Se on UCL The Bartlett School of Architecturen tutkimuksen pohjalta syntynyt tiedonjakoalusta, joka mahdollistaa algoritmiavusteiselle suunnitteluprosessille ominaisen suunnittelutiedon vaiheittaisen kehittymisen. Sen avulla pienemmistä osakokonaisuuksista voidaan helposti muokata ja yhdistellä tarkoituksenmukaisia suunnittelukokonaisuuksia. Speckle on hyvin uusi lisäosa, mutta sitä kehitetään avoimen lähdekoodin menetelmällä, minkä vuoksi kehitys on ollut nopeaa ja käyttäjien tarpeisiin joustavasti mukautuvaa. Sen tarkoituksena on mahdollistaa algoritmipohjainen suunnitteluprosessi, jossa suunnittelutieto linkittyy älykkäästi sekä suunnitteluosapuolien että suunnittelumallien välillä. (Speckle Works)

Speckle-lisäosaa voidaan hyödyntää sekä Rhinon että Grasshopperin käyttöliittymässä. Molemmissa käyttöliittymissä perusperiaate on sama: haluttua tietoa voidaan joko lähettää pilvipalveluun tai vastaanottaa sitä sieltä. Käyttäjän tulee rekisteröityä ja kirjautua omilla tunnuksillaan Speckleen, jotta hän voi käyttää palvelua. Specklellä on tällä hetkellä oma serveri, jota voi hyödyntää tiedon varastoimiseen, mutta myös itsemääritetyn serverin käyttäminen on mahdollista. Tässä tutkimuksessa käytetään ainoastaan Specklen

omaa serveriä. Pilvipalveluun ladattava suunnittelutieto on salattua ja sen voi määrittää olevan joko yksityistä tai julkista. Yksityiseen suunnittelutietoon voi lisätä oikeudet myös haluamilleen henkilöille, esimerkiksi kaikille projektiin osallistuville suunnitteluosapuolille. Speckle mahdollistaa näin niin sanotun multi-user työskentelyn, jossa useat eri osapuolet voivat muokata ja linkittää suunnittelutietoa keskenään reaaliaikaisesti. (Speckle Works)

Kun suunnittelija haluaa esimerkiksi jakaa geometriatietoa useisiin eri malleihin ja yhdistää nämä geometriat yhdeksi yhdistelmämalliksi, onnistuu se helposti Specklen avulla. Käytännössä haluttu suunnitteludata varastoidaan ensin Specklen määrittämän ID-tunnuksen ja itse määritetyn nimen avulla halutulle serverille. Tämän jälkeen haluttu suunnittelutieto voidaan hakea nimen tai tunnuksen avulla serveriltä ja hyödyntää sitä normaaliin tapaan. Kun alkuperäinen suunnittelutieto muuttuu, päivittyvät tiedot automaattisesti myös kaikkiin niihin malleihin, joissa tietoa hyödynnetään. Specklen komponentit toimivat dynaamisesti siten, että kun tiedon lähettävään komponenttiin lisätään uusi syöttöportti, muodostuu myös tiedon vastaanottaviin komponentteihin automaattisesti uusi tulostusportti samalla nimellä kuin syöttöportti. Tämä lisää tiedonsiirron selkeyttä ja helpottaa suunnitteluprosessin etenemistä erillisten algoritmimallien välillä.

Kuvassa 37 esitetään esimerkki Specklen toiminnasta. Siinä kuvataan kuvitteellinen tilanne, jossa kolme eri suunnittelijaa luovat kolme erilaista tietomallia siten, että suunnittelutieto päivittyy Specklen pilvipalveluun. Tämän jälkeen neljäs osapuoli kokoaa tietomallit yhteen ja jakaa muodostuneen yhdistelmämallin myös Specklen avulla valitulle serverille. Kaikki nämä neljä suunnittelijaa voivat työskennellä samanaikaisesti omien tiedostojensa parissa ja tieto linkittyy toisiinsa Specklen kautta automaattisesti.



Kuva 37. *Esimerkki suunnitteluosapuolien välisestä pilvipohjaisesta tiedonsiirrosta Specklen avulla.*

Edellä esitettyjen viiden lisäosan lisäksi tutkimuksessa käytetään ja kokeillaan myös lukuisia muita sovelluksia. Monet lisäosat koetaan tutkimuksessa hyödyllisiksi, mutta niiden vaikuttavuus ei ole työn kannalta merkittävä. Monien lisäosien soveltuvuus koetaan myös tämän tutkimuksen kannalta heikoksi, minkä vuoksi niiden toimintaperiaatetta ei avata tässä kappaleessa.

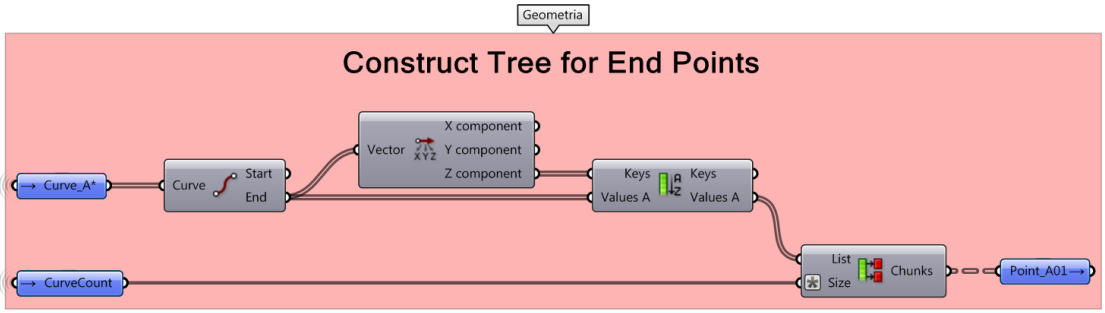
4.2.2 Tiedonsiirto

Koska algoritmiavusteinen suunnitteluprosessi perustuu toisiinsa linkitettyyn suunnittelutietoon, on tärkeää kiinnittää huomiota linkityksien oikeanlaiseen muodostamiseen. Erilaiset suunnittelukokonaisuudet linkittyvät toisiinsa aina jonkinlaisen rajapinnan kautta. Algoritmiavusteisessa suunnitteluprosessissa rajapinnat voidaan jakaa karkeasti kahteen perustyyppiin: tiedoston sisäiseen sekä kahden tiedoston väliseen rajapintaan. Tiedoston sisäisenä rajapintana voidaan pitää visuaalisen ohjelmointialustan avulla muodostettujen algoritmien sisäisiä linkityksiä. Laajempia algoritmimalleja käytettäessä tietosisältöä on hyvä jakaa pienempiin kokonaisuuksiin mallin sisällä. Tällöin osakokonaisuuksien välillä voidaan kuvitella sijaitsevan rajapinta. Algoritmisen suunnittelutiedon ominaisuuksista johtuen algoritmimallien sisäisten rajapintojen yksiselitteinen määrittäminen on haastavaa. Määritys perustuu käytännössä algoritmimallin sisällön luokitteluun esimerkiksi tekijän, suunnittelukokonaisuuden tai suunnittelutiedon ominaisuuksien mukaan.

Tässä tutkimuksessa algoritmimallien sisältöä pyritään jakamaan ja ryhmittelemään selkeiksi kokonaisuuksiksi mallin käytön helpottamiseksi. Tarkoituksenmukaisista kokonaisuuksista muodostetaan Grasshopperissa ryhmä (engl. group), joka nimetään sekä värikoodataan nimen mukaisesti. Tähän hyödynnetään MetaHopper-lisäosaa, joka mahdollistaa ryhmän automaattisen ja ennalta määritetyn värikoodauksen sille annettavan nimen mukaisesti. Eri väreillä kuvattuja kokonaisuuksia ovat esimerkiksi geometria, parametrit, Tekla ja RFEM. Värikoodauksen lisäksi ryhmälle annetaan kuvaava otsikko, joka kertoo siihen kuuluvien komponenttien suorittaman perustehtävän. Värien ja otsikoiden avulla mallin osakokonaisuuksien tunnistamista sekä niiden välisten rajapintojen havaitsemista voidaan helpottaa huomattavasti.

Värien ja otsikoiden lisäksi ryhmien toimintaa pyritään selkeyttämään kokoamalla aina kaikki ryhmään tulevat syötteet vasempaan reunaan ja tulosteet oikeaan reunaan. Näin ollen mallin sisäisten rajapintojen toimintaperiaate on helpommin tarkasteltavissa. Tämä voidaan toteuttaa Grasshopperin omilla komponenteilla, jotka kuvaavat tiedon primitiivisen muodon ja mahdollistavat tiedon nimeämisen. Yksi esimerkki tästä esitetään kuvassa 25. Toinen mahdollinen tapa on käyttää Telepathy-lisäosaa. Se on hyödyllinen lisäosa, joka sisältää ainoastaan kaksi komponenttia: etälähettäjän (engl. remote sender) ja etävastaanottajan (engl. remote receiver). Käyttäjä voi luoda ja nimetä ryhmän loppuun etälähettäjän. Tämän jälkeen tuplaklikkaamalla lähettäjää saadaan käyttöön etävastaanottaja, joka vastaanottaa tietoa sille annettavan nimen mukaan. Vastaanottajan voi nimetä itse tai

valita listasta, josta löytyvät kaikki jo käytetyt nimet. Vastaanottajaan voidaan ohjata tietoa myös useista lähteistä käyttämällä nimessä kerroinmerkkiä (*). Kuvassa 38 esitetään esimerkki, jossa eräs algoritmimallin sisältämä ryhmä on nimetty, värikoodattu ja otsikoitu sen sisällön mukaan. Lisäksi ryhmään tulevien syötteiden ja sieltä lähtevien tulosteiden hallinnoinnissa on hyödynnetty Telepathy-lisäosaa.



Kuva 38. *Esimerkki algoritmin jäsentelystä, otsikoinnista ja värikoodauksesta.*

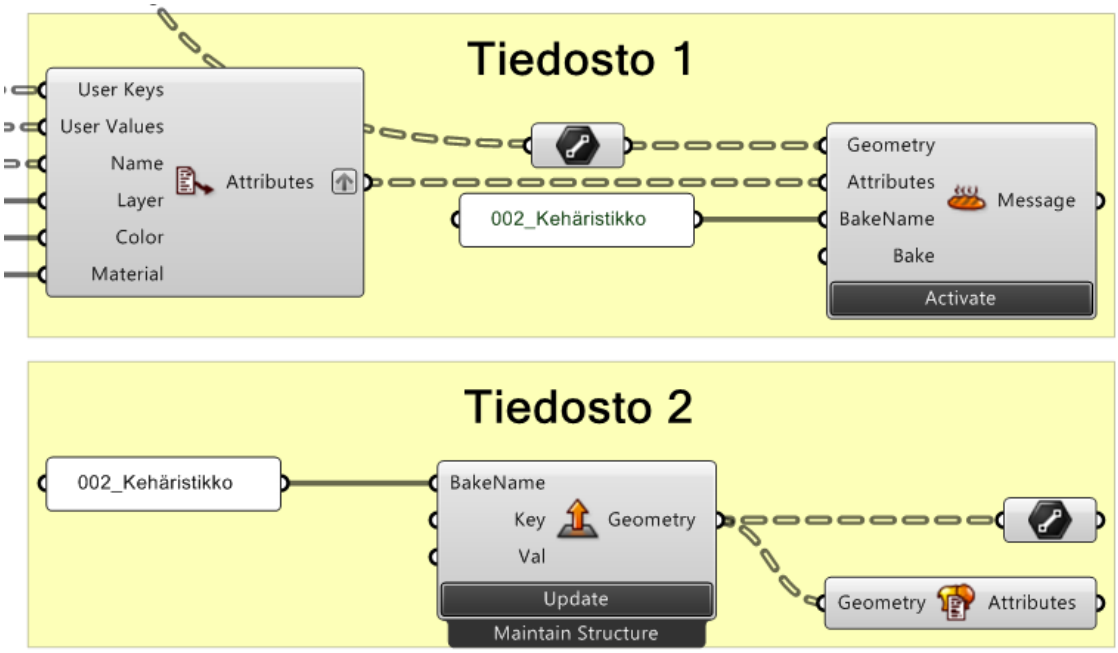
Toinen algoritmiavusteisessa suunnittelussa käytettävä tiedonlinkitystapa on kahden erillisen tiedoston välinen tiedonsiirto. Tiedonsiirto voi tapahtua joko kahden algoritmimallin tai algoritmimallin ja jonkin perinteisen suunnittelutiedon, esimerkiksi BIM-mallin, välillä. Algoritmimalli linkittyy käytettävään suunnitteluohjelmistoon tyypillisesti siten, että algoritmeihin tehtävät muutokset päivittyvät reaaliaikaisesti linkitettyyn suunnittelutietoon. Algoritmien ja ulkopuolisten ohjelmistojen välinen rajapinta kannattaakin usein sulkea erityisesti suuria tiedostoja muokatessa. Tällöin algoritmeihin tehtävät muutokset eivät päivitä linkitettyä tiedostoa automaattisesti ja muokkaaminen säilyy nopeana. Rajapinta voidaan tyypillisesti sulkea helposti asettamalla yksi tarvittava arvo todesta epätodeksi. Tutkimuksessa huomataan, että joidenkin lisäosien, kuten esimerkiksi Mathcad-ohjelmaa ohjaavan Macawin kohdalla, rajapinnan auki jättäminen saattaa aiheuttaa erityisiä ongelmia. Jos rajapinta on asetettu auki algoritmimallia avatessa, saattaa se kaataa koko ohjelman. Tällaisten tilanteiden estämiseksi tutkimuksessa hyödynnetään Grasshopper-lisäosaa, jonka avulla rajapinnan voi määrittää sulkeutuvaksi automaattisesti aina kun algoritmimalli avataan.

Täysin algoritmisessa suunnitteluprosessissa kaikki tarvittava suunnittelutieto sijaitsee algoritmimallissa eikä sillä ohjattavaa perinteistä suunnittelumallia ole välttämätöntä tallentaa ollenkaan. Esimerkiksi algoritmien avulla ohjattava FEM-malli voidaan luoda ja laskea sekä laskennan tulokset palauttaa algoritmimalliin kokonaan algoritmien avulla. Tällöin tiedostokooltaan suurta FEM-mallia ei ole välttämätöntä tallentaa ollenkaan, joskin se on suunnittelun raportoinnin kannalta suositeltavaa. Prosessin kannalta on usein luontevaa, että osa suunnittelusta tehdään algoritmiavusteisesti ja osa perinteisiä menetelmiä käyttäen. Tällöin tulee kiinnittää erityistä huomiota ulkopuolisia ohjelmia ohjaavien lisäosien toimintaan. Useimmissa lisäosissa voidaan esimerkiksi valita poistaako komponentti aiemmin luodut tiedot vai ei. Jos algoritmien ohjaamaa mallia muokataan perinteisin keinoin, tulee tämä huomioida rajapinnan asetuksia määrittäessä. Esimerkiksi

Tekla Live Linkin kautta mallia muokatessa rajapinta säilyttää aina kaikki BIM-mallissa luodut osat. Tästä syystä algoritmien avulla generoitavia osia ei tyypillisesti kannata tallentaa Tekla-malliin ennen kuin niiden algoritminen muokkaaminen on täysin valmis.

Kuten edellä esitetään, algoritmimallit linkittyvät usein johonkin perinteiseen suunnittelumalliin tai -tiedostoon jonkin Grasshopperin lisäosan avulla. Toinen oleellinen tiedonsiirtorajapinta sijaitsee kahden algoritmimallin välillä. Koska suunnittelussa hyödynnettäviä algoritmeja on hyvä jakaa kokonaisuuden selkeyttämiseksi pienempiin osiin, tulee myös algoritmimallien väliseen tiedonsiirtoon kiinnittää huomiota. Kahden algoritmimallin linkittäminen voidaan toteuttaa usealla tavalla, mutta tässä tutkimuksessa keskitytään tarkastelemaan kahta erilaista tapaa: Rhinoceros-ohjelman kautta tapahtuvaa sekä pilvipohjaisesti toteutettavaa tiedonsiirtoa. Koska Grasshopper on Rhinon lisäosa, yksi luotettava tapa siirtää tietoa kahden algoritmimallin välillä on muuntaa tietosisältö pysyväksi (engl. bake) Rhinossa. Tämä tapahtuu käytännössä siten, että Grasshopperin avulla luotavien algoritmien tulosteet, esimerkiksi geometriatieto, tallennetaan Rhino-tiedostoon ja tätä tiedostoa käytetään seuraavan algoritmimallin lähtötietona. Rhino-tiedostoon voidaan liittää myös erilaista metadataa ja ominaisuustietoa, jota voidaan hyödyntää seuraavan algoritmimallin luomisessa.

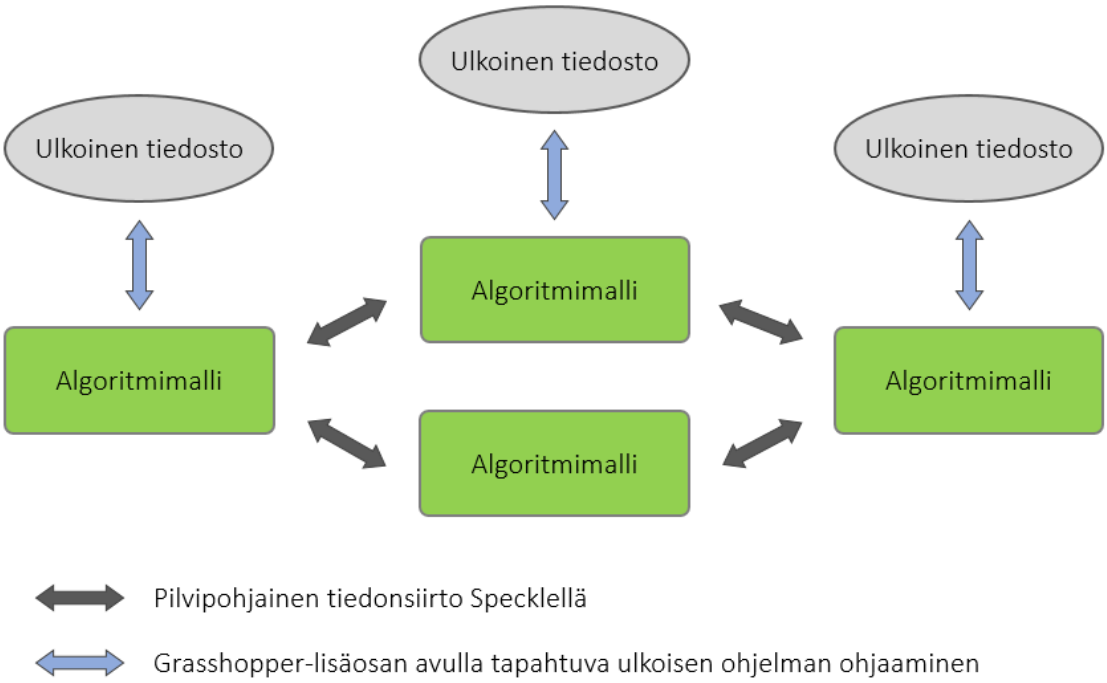
Rhinon kautta tapahtuvaan tiedonsiirtoon on kehitetty useita erilaisia lisäosia, joista yksi usein käytetty on Elefront. Sen perustehtävä on luokitella suunnittelutietoa Grasshopperin puolella siten, että tiedot siirtyvät myös Rhino-tiedostoon. Elefrontin avulla geometriatietoon voidaan liittää erilaisia ominaisuustietoja, kuten esimerkiksi nimi, tunnus, väri ja materiaali. Tämän jälkeen haluttu suunnittelutieto voidaan tallentaa pysyvästi Rhino-tiedostoon niin, että myös kaikki ennalta määritetyt ominaisuustiedot tallentuvat. Tallennettavalle suunnittelutiedolle tulee antaa nimi, jonka avulla siihen voidaan myöhemmin viitata. Kun suunnittelutieto on tallennettu Elefrontin avulla Rhinoon ja Rhino-tiedosto tallennettu, voidaan tätä tiedostoa käyttää tiedonsiirtoon. Rhino-tiedostosta on mahdollista saada kaikki tarvittava suunnittelutieto ilman, että niiden generoimia algoritmeja tarvitsee jakaa. Suunnittelija voi avata Rhino-tiedoston ja viitata siihen tallennettuun suunnittelutietoon aiemmin määritetyn nimen tai tunnuksen mukaan ja saada näin suunnittelutiedon käyttöön myös Grasshopperissa. Kuvassa 39 esitetään esimerkki siitä, miten Elefrontin avulla halutulle geometriatiedolle syötetään ensin ominaisuustiedot ja tämän jälkeen suunnittelutieto muunnetaan pysyväksi Rhinossa ja sille annetaan tunnuksena toimiva nimi. Tämän jälkeen suunnittelutietoon voidaan viitata aiemmin määritetyn nimen avulla.



Kuva 39. *Esimerkki algoritmimallien välisestä tiedonsiirrosta Elefrontin avulla.*

Toinen tässä työssä tutkittava ja testattava algoritmimallien välinen tiedonlinkitystapa on pilvipohjainen tiedonsiirto. Siihen kokeillaan kolmea erilaista Grasshopperin lisäosaa: Flux, Konstru ja Speckle. Näistä kolmesta päädytään lopulta käyttämään Speckleä. Specklen avulla tiedonsiirto voidaan toteuttaa suoraan Grasshopper-tiedostosta toiseen ilman erillistä siirtotiedostoa. Specklen avulla on helppo siirtää kaikenlaista tietoa ilman lisäosan ylimääräisiä sisäisiä luokituksia. Kun suunnittelutieto siirtyy erillisen serverin kautta, säästytään manuaaliselta siirtotiedostojen lähettämiseltä. Pilvipohjaisen tiedonsiirron avulla algoritmiavusteinen suunnitteluprosessi toimii huomattavasti automatisoidummin ja helpommin kuin Rhino-tiedostojen kautta tapahtuvassa tiedonsiirrossa. Speckleä käytettäessä viimeisin algoritmien luoma suunnittelutieto tallentuu aina käytettävälle serverille. Näin ollen, vaikka suunnittelun lähtötietona toimiva algoritmimalli suljetaan, pystytään sen pilvipohjaisesti varastoimaa tietosisältöä yhä muokkaamaan. Toisaalta kun algoritmimallia jälleen muokataan, päivittyvät muutokset reaaliaikaisesti pilvipalveluun ja sieltä tietoa hyödyntäviin algoritmimalleihin.

Tutkimuksessa kokeillaan erilaisia tiedonsiirtotapoja ja -ohjelmia, joiden avulla algoritmiavusteinen suunnitteluprosessi voidaan toteuttaa. Työssä päädytään siihen tulokseen, että pilvipohjainen tiedonsiirto on käytännöllisin tapa jakaa tietoa eri algoritmimallien välillä. Algoritmimallit pyritään jaottelemaan siten, että yksi algoritmimalli on yhteydessä korkeintaan yhteen ulkoiseen ohjelmaan. Kuvassa 40 esitetään periaatekuva tässä tutkimuksessa käytettävästä algoritmiavusteisen suunnitteluprosessin tiedonsiirrosta.



Kuva 40. *Periaatekuva tutkimuksessa tapahtuvasta tiedonsiirrosta.*

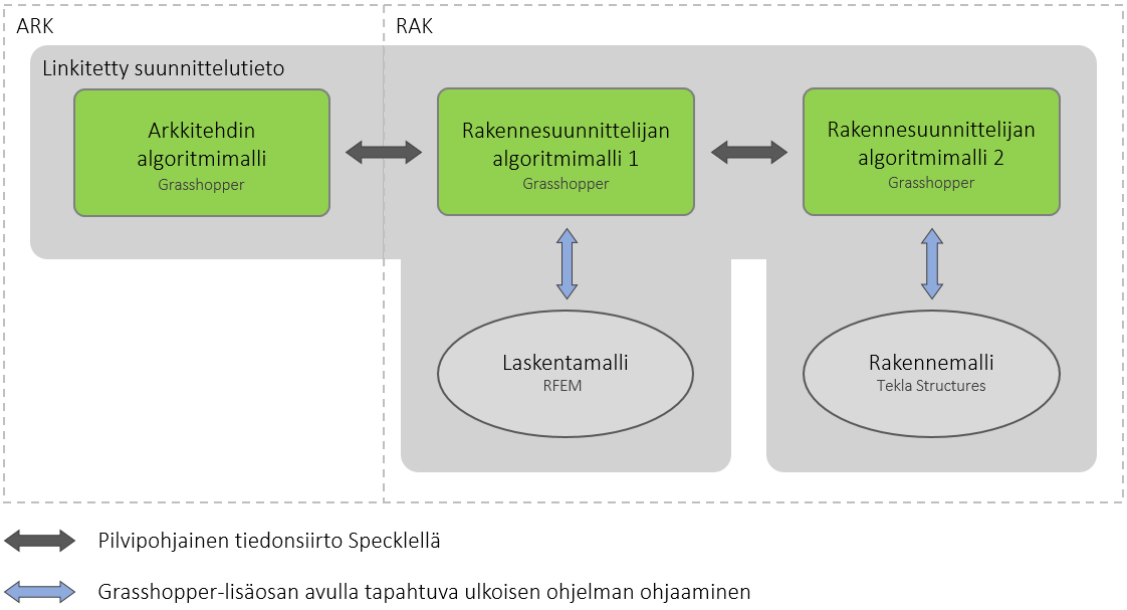
Erillisten mallien ja tiedostojen välisten rajapintojen lisäksi kiinnitetään erityistä huomiota algoritmimallien sisäiseen tiedonsiirtoon. Vaikka algoritmimallit pyritään säilyttämään maltillisen kokoisina, on niiden sisältöä hyvä jakaa vielä pienempiin osakokonaisuuksiin. Kun osakokonaisuudet nimetään, ryhmitellään ja visualisoidaan selkeästi, säilyvät algoritmimallit käyttökelpoisina myös tulevaisuudessa.

4.3 Teräsrakenteiden algoritmiavusteinen suunnittelu

Tässä tutkimuksessa keskitytään tutkimaan erityisesti teräsrakenteiden algoritmiavusteista suunnittelua. Suunnittelu koostuu kolmesta vaiheesta: mitoittamisesta, tietomallintamisesta ja raportoinnista. Tässä työssä tarkastellaan pääasiassa teräsrakenteiden mitoittamista sekä tietomallintamista. Rakenteiden mitoituksessa hyödynnetään RFEM-laskentaohjelmaa ja tietomallintamisessa Tekla Structures -ohjelmaa. Suunnittelun lähtötietona toimivat arkkitehdilta saatavat algoritmimallit, joista saadaan erityisesti rakenteiden päägeometria. Arkkitehdin ja rakennesuunnittelijan välinen tiedonsiirto toteutetaan myös Speckle-lisäosan avulla, jolloin koko arkkitehti- ja rakennesuunnittelun sisältävä prosessi toimii algoritmiavusteisesti. Näin ollen kaikki suunnittelutieto on linkitetty yhteen systeemiin koko prosessin ajan ja mahdollisten muutosten generoiminen onnistuu algoritmien avulla.

Rakennesuunnittelun lähtötietona toimivat arkkitehdin algoritmit ja niiden muodostama geometriatieto. Aluksi rakennesuunnittelija luo arkkitehdin geometriatiedon pohjalta laskentamallin. Laskentamalli luodaan RFEM-laskentaohjelmaan GH-RFEMLink-lisäosaa

hyödyntämällä. Laskentamallin avulla saadaan määritettyä rakenteiden sisäiset kuormitukset sekä kuormat rakenteellisesti kestävät teräsprofiilit. Kun rakenteiden mitoitus on suoritettu ja kestävyys varmistettu, voidaan saatujen tietojen pohjalta luoda rakennemalli. Rakennemalli luodaan Tekla Live Link -lisäosaa käyttämällä Tekla Structures -ohjelmaan. Rakennemallin luomisessa hyödynnetään sekä arkkitehdilta saatua geometriatietoa että laskentamallista saatavia poikkileikkaustietoja. Kuvassa 41 esitetään suunnitteluprosessin osakokonaisuudet sekä niissä hyödynnettävät ohjelmistot.



Kuva 41. *Teräsrakenteiden suunnittelussa hyödynnettävä algoritmiavusteinen suunnitteluprosessi.*

Kuten kuvasta 41 nähdään, tässä tutkimuksessa käytettävä algoritmiavusteinen rakennesuunnitteluprosessi koostuu kahdesta algoritmimallista sekä kahdesta perinteisestä suunnittelumallista. Tämän lisäksi rakennesuunnittelun lähtötietona toimii arkkitehdin algoritmimalli. Kaikki suunnittelutieto on linkitettynä toisiinsa, jolloin manuaalista tiedonsiirtoa ei tarvita. Suunnitteluprosessi on täysin algoritminen, jolloin kaikki suunnittelutieto luodaan ja muokataan algoritmien avulla. Ulkopuoliset ohjelmistot, RFEM ja Tekla, toimivat täten ainoastaan suunnittelutiedon havainnollistamisen sekä analysoimisen työkaluna. Tällöin tietokoneen rooli muuttuu tiedon esittäjästä tiedon generaattoriksi.

4.3.1 Teräsrakenteiden algoritmiavusteinen mitoitus

Teräsrakenteiden algoritminen mitoittaminen alkaa arkkitehdilta saatavan geometriatiedon analysoimisella ja muokkaamisella. Tyypillisesti geometria koostuu esimerkiksi viivoista, pinnoista ja tilavuuksista. Tässä tutkimuksessa arkkitehdin suunnittelema kattorakenne koostuu kaarevasta pinnasta sekä pinnan muodon mukaan luodusta viivageometriasta. Rakennesuunnittelijan työn kannalta oleellisinta on viivageometrian tietorakenne. Laskentamallin kannalta rakenteen muodostavat viivat tulee luokitella ja jaotella omiin

ryhmiinsä. Tämä tarkoittaa käytännössä geometriaobjektien järjestämistä puumuotoon Grasshopperissa. Arkkitehdin luoma viivageometria on usein jo valmiiksi puumuodossa, mutta rakennesuunnittelijan tulee tyypillisesti muokata geometriatieto oman työnsä kannalta soveltuvaksi kokonaisuudeksi. Käytännössä geometriatieto pyritään järjestelemään puumuotoon siten, että puurakenteen yhdellä oksalla on aina perusmuodoltaan samanlaiset rakenneosat. Kun arkkitehdin ja rakennesuunnittelijan välisessä tiedonsiirrossa käytetään Speckleä, suunnittelutieto säilyttää sille asetetun puumuodon. Suunnittelun kannalta on kuitenkin suotavaa, että Specklen kautta tapahtuvassa tiedonsiirrossa jokaisella oksalla sijaitseva suunnittelutieto syötetään omaan syöttöporttiinsa. Tällöin tiedon vastaanottavan tahon on huomattavasti helpompaa havaita tietorakenteessa tapahtuvat muutokset ja muokata omat algoritminsa muuttuneeseen tietoon soveltuvaksi.

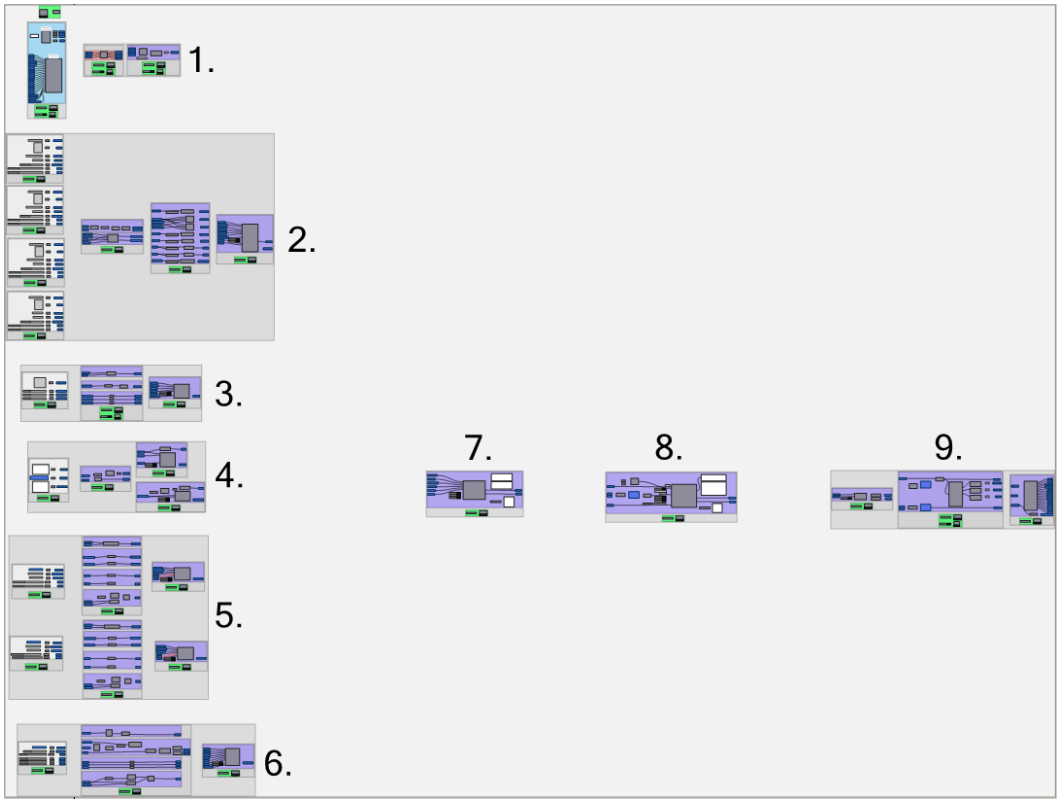
RFEM-laskentamallin luovaan GH-RFEMLink-lisäosaan on tarkoitus syöttää tietoa pääasiallisesti yksiulotteisessa puurakennemuodossa, jossa kullekin oksalle syötetään yksittäinen tieto tai tietolista. Kun suunniteltavien rakenteiden geometriatieto on ryhmitelty tarkoituksenmukaiseen puumuotoon, tulee kaikki siihen liitettävä ominaisuustieto syöttää geometriaa vastaavassa tietorakenteessa. Suunnittelijan tulee olla erityisen tarkkana komponenteille syötettävän tiedon rakenteesta, jotta linkki generoi laskentamallin oikein.

Kun arkkitehdin suunnittelema rakennekokonaisuus on analysoitu, voidaan laskentamallin luominen aloittaa GH-RFEMLink-lisäosan avulla. Laskentamalliin linkittyvä algoritmimalli voidaan jakaa yhdeksään perusosaan:

1. Geometriatiedon muokkaaminen ja ryhmittely.
2. Rakenneosien määrittäminen.
3. Ulkoisten tukien määrittäminen.
4. Kuormitustapausten ja -yhdistelyjen määrittäminen.
5. Viivakuormien määrittäminen.
6. Pistekuormien määrittäminen.
7. Laskentamallin luominen ja laskeminen.
8. Rakenneosien mitoittaminen ja optimointi.
9. Laskentatulosten analysointi.

Osien 1–6 avulla määritetään ensin laskentamallin ominaisuudet. Osia 5 ja 6 ei ole välttämätöntä määrittää, jos malli ei sisällä lainkaan tietyn tyyppisiä kuormia. Tämän jälkeen laskentamalli luodaan sille määritettyjen asetusten mukaisesti osassa 7. Kun laskentamalli on luotu onnistuneesti, voidaan suorittaa myös mallin laskenta. Seuraavaksi osassa 8 suoritetaan valitettujen rakenneosien Eurokoodin mukainen mitoitus. Tässä osassa on mahdollista myös optimoida rakenneosien poikkileikkaukset niiden käyttöasteen mukaan. Lopuksi kun laskentamalli on luotu, laskettu ja rakenneosat mitoitettu, voidaan laskennan tuloksena saadut tiedot palauttaa takaisin algoritmimalliin. Osassa 9 valitaan siis RFEM-mallista palautettavat tiedot, kuten esimerkiksi lopulliset poikkileikkaukset ja rakenteiden sisäiset kuormitukset. Näiden osien lisäksi algoritmimalli sisältää Specklen avulla tuote-

tun kokonaisuuden, joka vastaa algoritmimalliin tulevasta ja sieltä lähtevästä suunnittelutiedosta. Kuvassa 42 esitetään yleiskuva laskentamalliin linkitettävästä algoritmimallista.



Kuva 42. *Laskentamalliin linkitettävän algoritmimallin osat.*

Laskentamalliin linkitettävän algoritmimallin luominen aloitetaan jäsentelemällä arkkitehdin geometria rakennesuunnittelun kannalta sopivaan puumuotoon. Tämän jälkeen määritetään laskentamallin muodostavat rakenneosat. Rakenneosat luovalle komponentille syötetään lähtötiedoksi puumuotoinen viivageometria sekä samassa muodossa rakenneosien tyypit, päiden vapautukset, poikkileikkaukset, materiaalitiedot, rotaatiokulmat sekä nurjahduspituudet. Tämän jälkeen luodaan rakennejärjestelmän ulkoiset, pistemäiset tuennat. Tuet luovaan komponenttiin syötetään listana pisteet, joissa tuet sijaitsevat sekä tukien tyyppi ja asento. Seuraavaksi määritetään rakenteita kuormittavat kuormitustapaukset sekä kuormitustapauksista muodostuvat, laskennassa käytettävät kuormitusyhdistelyt. Lopuksi määritellään vielä rakenteisiin kohdistuvat kuormitukset. Kuormitukset luoviin komponentteihin syötetään kuormien suuruus, suunta ja tyyppi sekä pisteet ja rakenneosat, joihin kuormat kohdistuvat.

Kun kaikki edellä esitetyt kohdat on suoritettu, voidaan laskentamalli luoda RFEM:iin. Osissa 1–6 määritetyt osakokonaisuudet voidaan luoda laskentamalliin yksitellen omista komponenteistaan, mutta koko laskentamallin luominen onnistuu myös yhden napin painalluksella kohdassa 7. Laskentamallia luotaessa voidaan asetuksista valita, poistaako algoritmi kaiken laskentamallissa jo sijaitsevan tiedon vai ei. Kun tiedonsiirto aloitetaan

GH-RFEMLinkin avulla, RFEM suorittaa mallin luomisen Grasshopperissa asetettujen ehtojen mukaisesti. Kun laskentamalli on luotu, voidaan suorittaa mallin laskenta. Tämän jälkeen RFEM sisältää algoritmien avulla luodun laskentamallin sekä kaikki laskennassa saavutettavat tulokset.

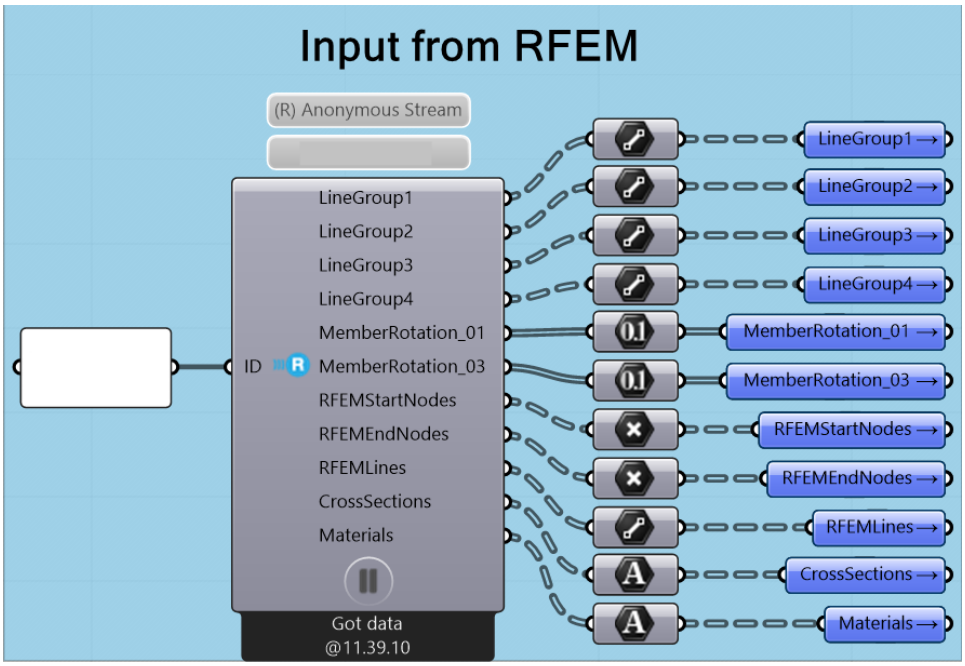
Laskentamallin luomisen ja ratkaisun jälkeen rakenneosat voidaan mitoittaa osassa 8. Siinä suoritetaan teräsosien mitoitus RFEM:n RF-STEEL EC3 -lisäosaa hyödyntämällä. Mitoituksen suorittavalla komponentilla voidaan myös tarvittaessa optimoida valitut poikkileikkaukset. Komponenttiin annetaan lähtötiedoiksi mitoitettavat rakenneosat, mitoituksessa käytettävät kuormitusyhdistelyt sekä laskennassa käytettävä Eurokoodin kansallinen liite. Lisäksi rakenteiden optimointia hyödynnettäessä, valitaan käytettävät poikkileikkaukset sekä tavoiteltavat käyttöasterajat. Kun rakenteiden mitoitus ja optimointi on suoritettu, voidaan varmistua rakennejärjestelmän toimivuudesta. Jos laskennan tulokset tyydyttävät eikä laskentamallia ole tarvetta muuttaa, voidaan algoritmimallin avulla tapahtuvassa suunnittelussa siirtyä osaan 9. Siinä on tarkoitus valita tulokset, jotka suunnittelija haluaa siirtää laskentamallista algoritmimalliin. Laskennan tuloksista voidaan palauttaa esimerkiksi optimoidut poikkileikkaustiedot sekä rakenneosissa vaikuttavat sisäiset kuormitukset. Laskentamallista saatavia tuloksia voidaan tämän jälkeen hyödyntää algoritmisessa suunnitteluprosessissa.

4.3.2 Teräsrakenteiden algoritmiavusteinen tietomallinnus

Laskentamallia luotaessa geometriatieto voidaan säilyttää melko yksinkertaisessa muodossa. Sauvamaisista rakenneosista muodostuva kattorakenne voidaan luoda vaivattomasti pisteiden ja viivojen avulla. Rakennemallia luotaessa geometriatiedon luominen ja muokkaaminen muuttuvat kuitenkin huomattavasti haastavammaksi. Enää rakenneosia ei voi ajatella vain yksiulotteisina viivoina vaan kaikilla osilla on jonkinlainen tilavuus. Koska rakennemallin tulee vastata todellisia rakenteita usein millimetrin tarkkuudella, saattaa sen luominen on olla joskus haastavaa. Tässä työssä suunniteltava kattorakenne on jo perusmuodoltaan melko vaativa, joten rakenteita ei pyritä suunnittelemaan ja mallintamaan täysin toteutuskelpoisiksi. Tutkimuksessa pyritään kuitenkin käymään läpi kattorakenteen rakenneosien tietomallinnuksen perusperiaatteet.

Kun rakennejärjestelmän rakenteellinen toimivuus on varmistettu ja teräsrakenteet mitoitettu, voidaan aloittaa rakennemallin teko. Rakennemalli luodaan algoritmiavusteisesti Tekla Structures -ohjelmaan Tekla Live Link -lisäosaa hyödyntämällä. Rakennemallin luominen aloitetaan muokkaamalla geometriatieto haluttuun muotoon. Tässä voidaan hyödyntää sekä arkkitehdilta saatavaa että laskentamallin luomiseen käytettävää geometriatietoa tai niiden yhdistelmää. Suunnittelussa tulee huomioida, että geometria vastaa riittävästi laskennassa käytettyä mallia. Tällöin voidaan varmistua järjestelmän ja sen osien rakenteellisesta toimivuudesta. Tässä työssä rakennemallin luomisessa hyödynnetään laskentamallin viivageometriaa, joka ainoastaan järjestetään arkkitehdin alkuperäistä

geometriaa vastaavaan puumuotoon. Lisäksi laskentamallista saatavat profiili- ja materiaalitiedot muokataan Tekla-mallin ymmärtämään muotoon. Rakennemallin lähtötietoina hyödynnettävät tiedot saadaan laskentamallin algoritmimallista Speckle-lisäosan avulla. Kuvassa 43 esitetään RFEM-laskentaohjelmaan linkitetystä algoritmimallista saatavat lähtötiedot.

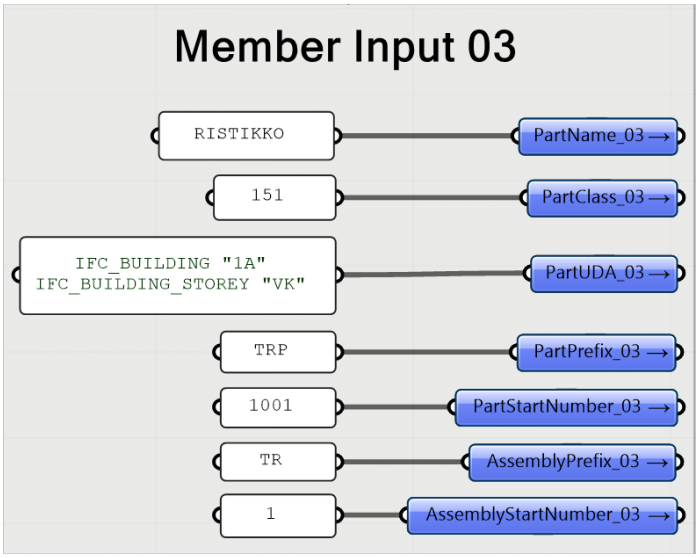


Kuva 43. *Rakennemallin luomiseen hyödynnettävät lähtötiedot.*

Kun geometria on jäsennelty haluttuun tietomuotoon, voidaan siihen tehdä rakennemallin kannalta tarvittavia muutoksia. Esimerkiksi teräsosien päätepisteiden sijaintia on usein tarpeen muokata. Laskentamallissa viivat kohtaavat toisensa pääasiassa samassa pisteessä ja kun tätä geometriaa hyödynnetään rakennemallin luomisessa, tarkoittaa se useiden rakenneosien välisiä törmäyksiä. Tästä syystä rakennemallia luotaessa on usein suotavaa muokata viivageometria enemmän todellisuutta vastaavaksi. Tekla Live Linkin avulla luotavia sauvamaisia rakenteita ja niiden päätepisteiden sijaintia voidaan muokata Tekla Live Linkin omien komponenttien avulla. Tässä tutkimuksessa koettiin kuitenkin helpommaksi muokata ensin alkuperäinen geometria haluttuun muotoon eikä Teklan Deforming Attributes -komponenttia näin käytetty.

Kun viivageometria on jäsennelty ja muokattu voidaan rakenneosat luoda Tekla Live Linkin komponenttien avulla. Tässä tutkimuksessa hyödynnettiin ainoastaan kahta komponenttia, jotka olivat yksinkertainen teräsrakenteille tarkoitettu Beam-komponentti sekä monimuotoisemmille sauvarakenteille tarkoitettu Extrude Beams -komponentti. Molempien komponenttien lähtötiedot ovat hyvin samanlaiset. Molempiin komponentteihin syötetään ensin geometriatieto: Beam lukee viivageometriaa ja Etrude Beams pisteitä. Tämän lisäksi rakenneosat luoviin komponentteihin syötetään osien attribuutti, sijainti ja numerointitiedot. Tässä tutkimuksessa osille annettiin nimi, profiili, materiaali, luokka, etuliite

sekä numerotunnus. Tämän lisäksi annettiin teräskokoonpanon etuliite, numerotunnus sekä IFC-tiedonsiirtoon tarvittavia lisätietoja, kuten rakennus- ja kerrostunnus. Kuvassa 44 esitetään rakenneosille algoritmimallissa syötettävät tiedot. Profiili- ja materiaalitieto saadaan suoraan laskentamallista.



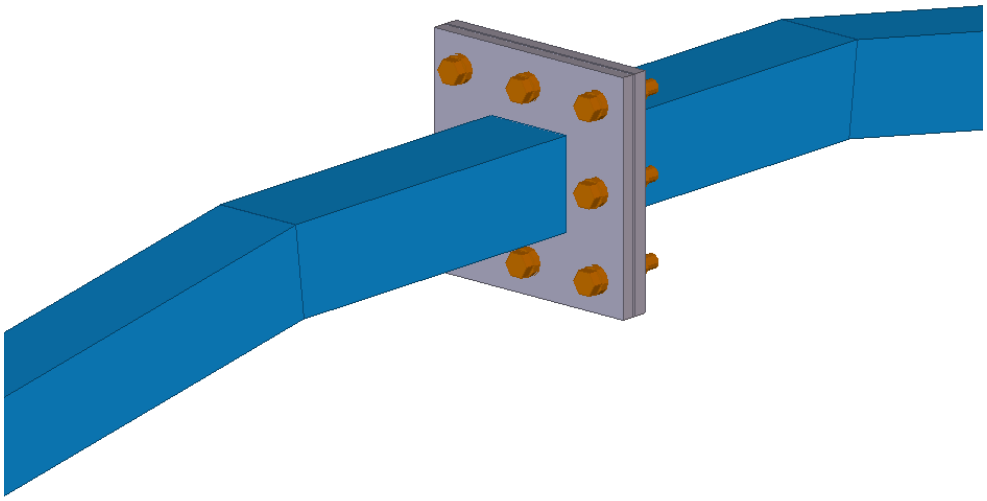
Kuva 44. *Rakennemallin rakenneosalle syötettävät lähtötiedot.*

Kun rakenneosat on luotu, tehdään niihin vielä joitakin muokkauksia. Tässä tutkimuksessa se tarkoittaa rakenneosien päiden leikkaamista haluttuun muotoon sekä osien välisten hitsiliitosten määrittelyä. Tekla Structuresissa on hyödyllinen työkalu nimeltään Fitting. Sen tarkoituksena on leikata valittavat sauvat halutun tason suuntaisesti. Tekla Live Linkin avulla voidaan käyttää kyseistä Fitting-työkalua ja sen avulla teräsosien päät voidaan leikata haluttuun muotoon. Tämän avulla teräsosat eivät enää törmää toisiinsa. Kun rakenneosien päät on leikattu tarkoituksenmukaisesti, voidaan osat hitsata toisiinsa kiinni. Tekla Structuresissa hitsiliitokset määrittävät ennen kaikkea rakenneosien muodostamat kokoonpanot. Tätä tutkimusta tehtäessä Tekla Live Link ei sisällä vielä hitsiliitosten tekemiseen tarvittavaa komponenttia, joten hitsiliitokset toteutetaan tässä itsemääritetyn komponentin avulla.

Tämän jälkeen algoritmimallin avulla luotava rakennemalli on valmis. Se sisältää kaikki tarvittavat rakenneosat ja niiden ominaisuustiedot. Osat on määritetty siten, että ne eivät törmää keskenään ja halutuista osista on muodostettu teräskokoonpanoja hitsiliitosten avulla. Jos algoritmiavusteista suunnittelua ei tarvitse enää hyödyntää voidaan rakennemalli tallentaa ja esimerkiksi jatkaa sen käsittelyä perinteisin menetelmin. Kun rakennemallin osat on luotu Tekla Live Linkin avulla, voidaan suunnittelua jatkaa yhä algoritmiavusteisesti. Tekla Live Link sisältää komponentin, jolla todelliset rakenneosat ja niiden tarkka muoto voidaan palauttaa takaisin Rhinon ja Grasshopperin ymmärtämään muotoon. Näin ollen rakennesuunnittelija voi muodostaa rakennemallistaan kolmiulotteiset osat sisältävän geometriatiedon ja jakaa tämän tiedon esimerkiksi arkkitehdeille.

4.4 Jatkosliitoksen algoritmiavusteinen mitoitus ja mallinnus

Algoritmiavusteista suunnittelua voidaan hyödyntää myös liitosten mitoituksessa ja tietomallinnuksessa. Tässä tutkimuksessa liitossuunnitteluprosessia tutkitaan yksinkertaisen jatkosliitoksen avulla. Liitos koostuu liitettävistä terässauvoista, niiden päihin hitsattavista päätylevyistä sekä levyt yhdistävästä pulttiryhmästä. Yhdistettävinä osina toimivat kattorakenteen putkiprofiilit. Jatkosliitos esitetään kuvassa 45.



Kuva 45. *Algoritmiavusteisesti suunniteltava jatkosliitos.*

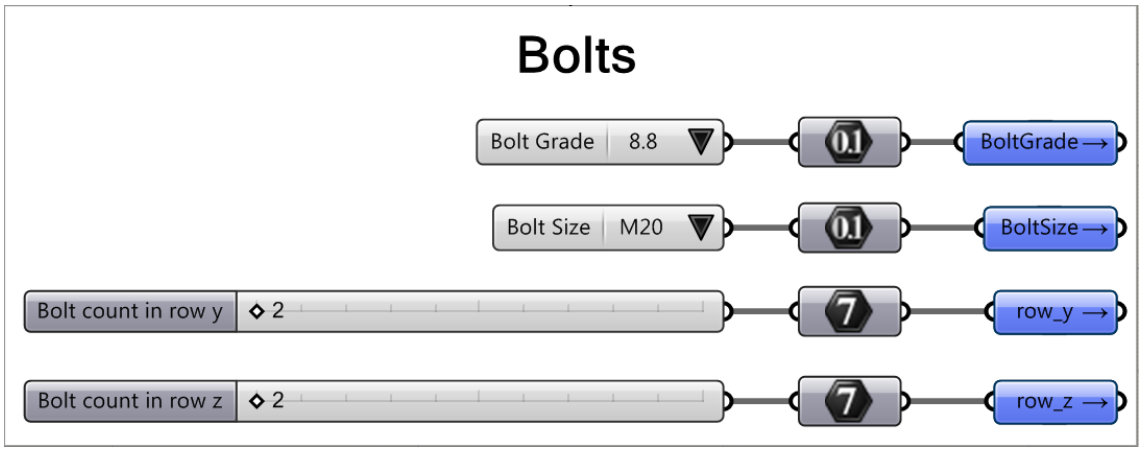
Liitoksen suunnittelu suoritetaan tässä tutkimuksessa täysin algoritmiavusteisesti. Liitossuunnittelu voidaan jakaa neljään pääosaan, joista jokaisessa käytetään omaa algoritmimallia. Ensimmäiseksi määritetään liitoskohdassa vaikuttavat kuormitukset. Tämä tapahtuu kappaleessa 4.3.1 esitettävän laskentamallin avulla. Tämän jälkeen kuormitustietoja jäsennellään tarkasteltavan liitoskohdan mukaisesti omassa algoritmimallissaan. Kolmannessa algoritmimallissa suunnitellaan liitos siten, että se kestää siihen kohdistuvat kuormitukset. Tässä hyödynnetään Mathcad-laskentapohjaa ja sitä ohjaavaa lisäosaa Macawia. Lopuksi liitos mallinnetaan kappaleessa 4.3.2 esitettävään tietomalliin. Myös liitossuunnitteluprosessissa algoritmimallien välinen tiedonsiirto tapahtuu Speckle-lisäosan avulla.

Liitossuunnittelu alkaa liitoskohtaan kohdistuvien kuormitusten määrittämisellä. RFEM-laskentaohjelmaan linkitetystä algoritmimallista saadaan kaikkien rakenneosien päissä vaikuttavat kuormitukset. Näin ollen laskentamallissa tulee sijaita piste siinä kohdassa, johon suunniteltava liitos on tulossa. Laskentamalliin linkitettävästä algoritmimallista saadaan ulos kaikki rakennejärjestelmään vaikuttavat kuormitukset. Koska suunniteltava liitos sijaitsee ainoastaan tietyissä kohdissa rakennekokonaisuutta, tulee laskentatuloksia

rajata. Laskentatuloksia voidaan rajata jonkin verran laskentamallin luovassa algoritmimallissa, mutta lopullinen jäsentely ja valinta suoritetaan selvyyden vuoksi omassa algoritmimallissaan. Jos laskentamallista saatavan tulosdatan koko on suuri, saattaa se vaikeuttaa Specklen kautta tapahtuvaa tiedonsiirtoa. Tällöin tiedonsiirto voidaan suorittaa esimerkiksi Excel-taulukoiden avulla. Vaikka algoritmimallien välisessä tiedonsiirrossa käytetään tässä työssä ainoastaan Speckle-lisäosaa, toimii Excel hyvänä tiedon raportointialustana. Kaikki laskentamallista saatava tulostieto voidaan helposti siirtää Excel-taulukkoon esimerkiksi Bumblebee-lisäosan avulla.

Kun laskenta on suoritettu ja laskentatulosten esivalinta tehty, voidaan tulokset siirtää seuraavaan algoritmimalliin. Tulostieto koostuu rakennejärjestelmän muodostavista rakenneosista ja niiden välisistä pisteistä, pisteiden ja sauvojen metatiedoista sekä sauvojen alku- ja loppupisteissä vaikuttavista kuormituksista. Toisessa algoritmimallissa valitaan pisteet, joihin suunniteltavat liitokset tulevat ja tarkastellaan niiden kuormitustietoja. Koska RFEM:stä saatavassa tulostiedossa ovat ainoastaan sauvojen päissä vaikuttavat kuormitukset, tulee liitoskohtaan liittyvien sauvojen kuormitukset yhdistellä. Jos kuormitustapauksia ja -yhdistelmiä on paljon, voidaan liitoksen suunnittelussa käytettäviä kuormituksia rajata erilaisten kriteerien perusteella. Kun tulosdata on rajattu ja jäsennelty, voidaan se siirtää eteenpäin seuraavaan algoritmimalliin. Seuraavaksi tapahtuvaan liitossuunnitteluvaiheeseen siirretään liitettävien sauvojen profiilitieto sekä liitokseen kohdistuvat kuormitukset. Kuormitustieto koostuu normaalivoimasta sekä kahteen suuntaan vaikuttavasta leikkausvoimasta ja taivutusmomentista.

Algoritmiavusteisen liitossuunnitteluprosessin kolmannessa vaiheessa suoritetaan liitoksen mitoitus. Siinä valitaan liitoksen ominaisuudet siten, että se kestää siihen kohdistuvat kuormitukset. Liitossuunnittelu voidaan toteuttaa algoritmiavusteisten menetelmien avulla hyvin automatisoidusti, mutta suunnittelu voidaan jättää myös suunnittelijan omien päätöksien suhteen avoimeksi. Tässä tutkimuksessa ei pyritä ainoastaan kuormitustietoihin perustuvaan automatisoituun liitossuunnitteluun vaan suunnittelijalle jätetään vapaus muokata liitos itsemääritettyjen ominaisuustietojen mukaiseksi. Algoritmimallissa tapahtuva liitossuunnittelu alkaa liitoksen ominaisuuksien valinnalla. Suunnittelija määrittää päätylevyjen koon ja materiaalin, päätylevyjen ja sauvojen väliset hitsiliitokset sekä levyt yhdistävän pulttiryhmän ominaisuudet. Pultit sijoittuvat tasavälisiin riveihin putkiprofiilin ympärille. Kuvassa 46 esitetään esimerkki pulttiryhmän ominaisuuksien määrittelystä. Määritettävinä ominaisuuksina ovat pulttien lujuusluokka, koko ja määrä.

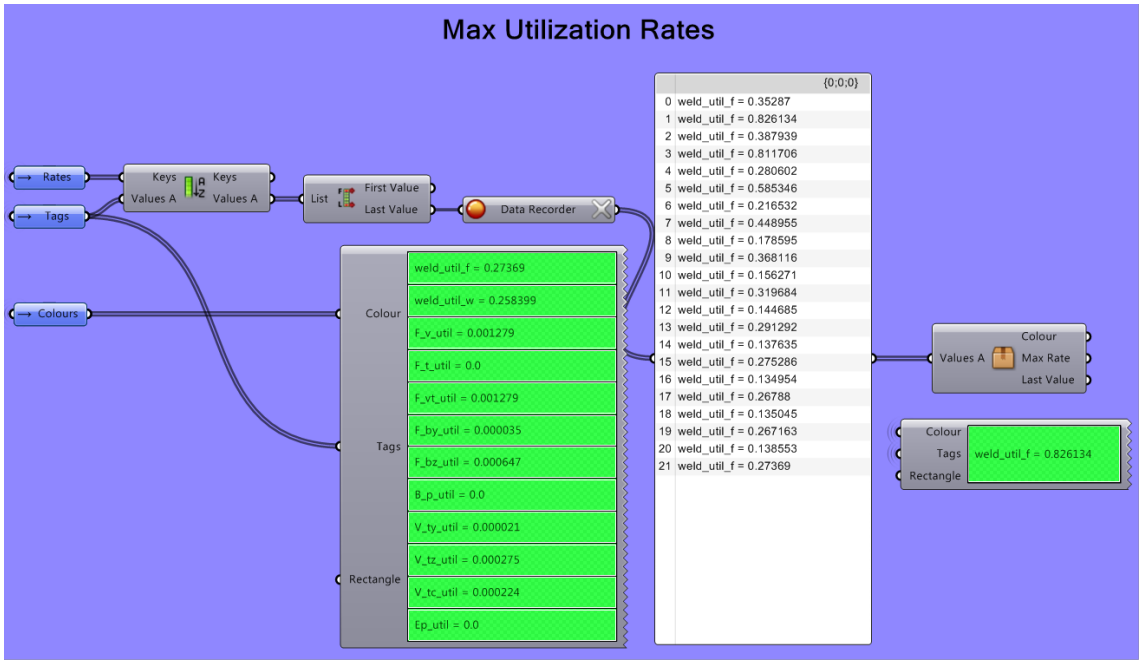


Kuva 46. *Jatkosliitoksen pulttiryhmän ominaisuuksien määrittely.*

Kun kaikki liitoksen ominaisuustiedot on määritetty Grasshopperissa, voidaan ne linkittää valmiiseen Mathcad-laskentapohjaan Macaw-lisäosan avulla. Laskentapohjaan määritetään kaikki Grasshopperissa luotavat syötteet sekä sinne palautettavat tulosteet. Tässä tutkimuksessa Macawin kautta siirretään yhteensä 25 syötettä laskentapohjaan. Jotta laskenta toimii oikein, tulee syötearvoihin liittää oikea tunnus sekä yksikkö. Kun laskentapohjaan on linkitetty kaikki halutut lähtöarvot, voidaan laskenta suorittaa. Kun laskenta on suoritettu, ennalta määritetyt tulosteet palautetaan takaisin Grasshopperiin. Tässä tutkimuksessa tulostetietoja on yhteensä 16 ja niistä 12 on liitoksen kestävyyteen liittyviä arvoja ja loput neljä liitoksen lopullista geometriaa kuvaavia arvoja. Koska Mathcadin avulla luodut laskentapohjat eivät tyypillisesti ole kovin raskaita, voidaan laskenta pitää koko ajan päällä. Tällöin laskennan alkuarvoja muutettaessa Mathcad-laskentapohja suorittaa laskennan reaaliaikaisesti ja palauttaa laskennan tulokset automaattisesti takaisin Grasshopperiin.

Grasshopperiin palautettavista tulosteista nähdään, kestääkö suunniteltava liitos vallitsevan kuormitustapauksen sellaisenaan. Koska määräävän kuormitustapauksen määrittäminen on haastavaa, tulee liitoksen kestävyyttä tarkastella erilaisten kuormitusolosuhteiden vallitessa. Tämä toteutetaan siten, että RFEM:stä saatavia eri kuormitusyhdistelyissä vallitsevia kuormituksia syötetään Mathcadiin Hoopsnake-lisäosan avulla. Hoopsnake-komponentti pystyy käyttämään, Grasshopperin perusperiaatteesta poiketen, lähtötietonaan sen omaa tulostetta. Tämä tarkoittaa sitä, että liitoksen mitoitukseen käytettävään Mathcad-laskentapohjaan voidaan syöttää kaikki halutut kuormitustapaukset automatisoidusti. Laskentapohja säilyttää kaikki muut syötteet samana ja muuttaa ainoastaan liitokseen kohdistuvia kuormituksia. Kun kaikki tarvittavat kuormitustapaukset on käyty lävitse, voidaan varmistua liitoksen kestävyydestä. Algoritmimalli tallentaa kaikkien kuormitustapausten määräävimmän murtumismuodon ja sen käyttöasteen. Määräävän murtumismuodon ja käyttöasteen avulla suunnittelija voi arvioida liitoksen toimintaa ja muokata sen ominaisuuksia haluttuun suuntaan. Kuvassa 47 esitetään esimerkki liitoksen kestävyystarkastelussa saatavista käyttöasteista. Algoritmi esittää kaikki tarkasteltavat

murtumismuodot ja niiden käyttöasteet vallitsevassa kuormitustapauksessa. Lisäksi algoritmi esittää listan kaikkien läpikäytyjen kuormitustapausten määräävimmistä murtumismuodoista ja niiden käyttöasteista sekä listan suurimman käyttöasteen.



Kuva 47. *Liitososien käyttöasteiden tarkastelu eri kuormitustapauksissa.*

Kun liitoksen kestävyys ja toimivuus on varmistettu, siirrytään suunnittelun viimeiseen vaiheeseen, tietomallintamiseen. Siinä suunniteltava liitos luodaan Tekla Live Linkin avulla käytettävään tietomalliin. Liitokset luodaan rakennemalliin samassa algoritmimallissa kuin muukin rakennemalli. Mallinnuksen lähtötietona käytetään laskennassa määritettyjä arvoja, kuten esimerkiksi päätylevyjen koko, pulttien määrä, koko ja sijainti sekä hitsiliitosten koko. Nämä tiedot tuodaan algoritmimalliin edellisestä mallista jälleen Specklen avulla. Lisäksi algoritmimallissa määritetään rakennemallin kannalta oleellisia tietoja, kuten liitososien nimi-, luokka- ja numerointitietoja. Liitosten mallinnukseen Teklassa voi käyttää erilaisia valmiita liitoskomponentteja, mutta se ei ole välttämätöntä. Tässä tutkimuksessa jatkosliitos toteutetaan mallintamalla erikseen päätylevyt sekä niiden välinen pulttiryhmä. Lisäksi sauvojen ja päätylevyjen väliset hitsiliitokset määritetään erikseen. Pulttiryhmän mallinnukseen käytetään Teklan omaa komponenttia Bolt Macro.

Jatkosliitoksen mallintaminen koostuu viidestä perusosasta. Ensimmäiseksi määritetään yhdistettävät osat. Kun liitoksen mallinnus tapahtuu samassa algoritmimallissa kuin rakenneosien mallinnus, on haluttujen rakenneosien valitseminen melko helppoa. Seuraavaksi luodaan rakenneosien päihin tulevat päätylevyt. Levyille määritetään laskennassa saadut arvot sekä muut ominaisuustiedot ja ne luodaan rakennemalliin. Tämän jälkeen liitettävien rakenneosien päät leikataan levyjen sijainnin mukaan Fitting-komponenttia

hyödyntämällä. Neljännessä vaiheessa päätylevy hitsataan kiinni rakenneosaan laskennassa määritetyllä hitsiliitoksella. Viimeisessä vaiheessa luodaan vielä liitoksen pultit Teklan pulttiryhmäkomponentin avulla. Pulttiryhmää varten tulee määrittää ennen kaikkea liitoksen sijaintiin, suuntaan ja muihin ominaisuuksiin vaikuttavat syötetiedot. Kun kaikki nämä vaiheet on suoritettu rakennemalli sisältää kaikki lujuusteknisesti varmennetut jatkosliitokset.

5. TUTKIMUSTULOSTEN ANALYSOINTI

Algoritmiavusteinen suunnitteluprosessi sisältää samoja osia ja vaiheita kuin perinteinenkin suunnitteluprosessi. Rakennejärjestelmät, rakenneosat ja liitokset tulee mitoittaa ja mallintaa aivan kuten perinteisessäkin prosessissa. Algoritmiavusteisessa suunnittelussa voidaan hyödyntää myös tuttuja suunnitteluohjelmistoja, kuten esimerkiksi RFEM-laskentaohjelmaa ja Tekla Structuresia. Vaikka perinteisellä ja algoritmisia menetelmiä hyödyntävällä suunnitteluprosessilla on monia samankaltaisuuksia, eroavat niiden toteutustavat selvästi toisistaan. Algoritmiavusteinen prosessi perustuu ennen kaikkea sääntöihin, joiden avulla suunnittelu toteutetaan. Sääntöinä toimivat erilaiset algoritmit, joita voidaan luoda ja muokata visuaalisen ohjelmointialustan avulla. Suunnittelu etenee algoritmien avulla luodun logiikan mukaan erilaisten suunnittelumallien sisällä sekä niiden välillä. Täysin algoritmisessa prosessissa kaikki suunnittelutieto on linkitettynä toisiinsa ja suunnittelu voi edetä automatisoidusti aina arkkitehdin lähtötiedoista lujuusteknisesti varmennettuihin, toteutuskelpoisiin rakenteisiin ja liitoksiin.

Tutkimuksen tavoitteena on käydä läpi rakennesuunnitteluprosessin päävaiheet ja tarkastella niiden ominaisuuksia. Algoritmiavusteisen suunnittelun luonteesta johtuen tutkimuksessa keskitytään tarkastelemaan erityisesti suunnittelutiedon vaiheittaista analysointia ja muokkausta sekä suunnitteluprosessin sisällä tapahtuvaa tiedonsiirtoa. Algoritmiavusteista suunnittelua hyödynnetään vielä melko suppeasti rakennesuunnittelussa, mutta sillä on suuri potentiaali kasvaa laajasti käytetyksi menetelmäksi. Algoritmiavusteinen suunnittelu on käsitteenä hyvin laaja ja sitä voidaankin käyttää monissa eri tehtävissä, eri asteisesti. Algoritmiset menetelmät eivät sovellu kaikkiin suunnittelussa suoritettaviin tehtäviin, joten tyypillisesti suunnitteluprosessi on vain osittain algoritmiavusteinen. Käytännön kokemuksen kautta on mahdollista parantaa tietämystä siitä, minkälaisissa tehtävissä algoritmisista menetelmistä saadaan lisäarvoa.

Tässä luvussa on tarkoitus analysoida tapaustutkimuksen sisältöä sekä siitä saatavia tuloksia. Koska tutkimuksessa tarkastellaan laaja-alaisesti koko rakennesuunnitteluprosessia, on aiheesta vaikea esittää numeerisia tuloksia. Tutkimuksen tuloksina voidaan pitää ennen kaikkea suunnitteluprosessin eri vaiheiden ja niissä hyödynnettävien työkalujen toimivuutta. Tässä luvussa käsitellään ensin algoritmiavusteisen suunnitteluprosessin toimivuutta kokonaisuutena. Prosessin toimivuus jaotellaan sen hyödynnettävyyteen ja käytettävyyteen sekä ongelmiin ja puutteisiin. Prosessin toimivuuden lisäksi luvussa analysoidaan algoritmiavusteisten työkalujen toimivuutta. Käytännön suunnittelussa käytettäviä työkaluja ja niiden toimivuutta tutkitaan niin mitoituksen, mallinnuksen kuin tiedonsiirronkin kannalta. Lopuksi luvussa tarkastellaan vielä algoritmiavusteisen suunnittelun jatkokehitystarpeita.

5.1 Algoritmiavusteisen suunnitteluprosessin toimivuus

Kun algoritmiavusteista suunnitteluprosessia aletaan tutkia ja kehittää lähes tyhjältä pöydältä, tulee tutkimuksessa vastaan monenlaisia haasteita. Joihinkin ongelmiin löytyy vastaus helposti ja, joidenkin asioiden ratkaiseminen on hyvin työlästä. Jotkut tekijät jäävät myös täysin ratkaisematta. Tutkimuksen tekeminen perustuu pääasiassa asioiden kokeiluun, tulosten analysointiin ja niihin reagoimiseen. Algoritmiavusteisen suunnitteluprosessin kehittäminen onkin jatkuva iteratiivinen prosessi, jossa pyritään asteittaisesti etenemään kohti paremmin toimivaa kokonaisuutta. Lopullista tilaa on käytännössä mahdotonta saavuttaa, koska aina löytyy jatkokehitystarpeita. Lisäksi rakennesuunnitteluprosessi on hyvin laaja ja monimutkainen kokonaisuus, jossa kaikkien yksittäisten tehtävien nimeäminen on haastavaa. Monia asioita voidaan suorittaa myös useilla eri tavoilla. Tästä syystä tutkimuksessa keskitytään pääasiassa laajempiin selvästi havaittaviin osakokonaisuuksiin sekä näiden osakokonaisuuksien välisiin vuorovaikutussuhteisiin.

Kuten kuvassa 28 esitetään, algoritmiavusteisessa suunnittelussa työvaiheiden kestot jakautuvat eri tavoin perinteiseen suunnitteluun verrattuna. Samaan tulokseen päästään myös tässä tutkimuksessa. Suurin eroavaisuus on siinä, että algoritmisessa suunnittelussa ensimmäiset käyttökelpoiset tulokset saadaan tyypillisesti myöhemmin kuin perinteisessä suunnitteluprosessissa. Toisaalta suunnittelussa ensimmäiseksi saavutettava ratkaisu harvoin on lopullinen vaan suunnitelmia pyritään kehittämään tehokkaammiksi. Kun suunnittelua jatketaan iteratiivisesti ja suunnitelmista muodostetaan vaihtoehtoisia ratkaisuja, algoritmiavusteinen suunnittelu muuttuu selvästi perinteistä prosessia nopeammaksi. Suunnitteluprosessien ero riippuu siis pääasiassa suunniteltavien rakenteiden ominaisuuksista ja lopullisen ratkaisun löytymiseen tarvittavien iteraatiokierrosten määrästä. Mitä enemmän suunnitelutietoa joudutaan muuttamaan ja mitä työläämpää muutosten toteuttaminen on, sitä kannattavampaa on tyypillisesti hyödyntää algoritmiavusteista suunnittelua.

Kun perinteisessä suunnitteluprosessissa suunnittelija luo ja muokkaa suunnittelutietoa esimerkiksi jossakin mallinnusohjelmassa, algoritmiavusteisessa suunnitteluprosessissa suunnittelija luo mallin generoivia algoritmeja. Algoritmien luominen saattaa olla, geometriasta riippuen, huomattavasti haastavampaa ja työläämpää kuin perinteisen laskenta- tai tietomallin luominen. Algoritmit kuitenkin sisältävät vaiheet, miten suunnittelu etenee ja miten suunnittelutieto muuttuu suunnittelun edetessä. Tällöin suunnittelussa voidaan helposti palata johonkin aikaisempaan suunnitteluvaiheeseen. Algoritmit ovat käytännössä aina parametrisia, jolloin niiden tulostetta voidaan muokata lähtötietoina toimivia parametreja muuttamalla. Algoritmeja luotaessa on hyvin tärkeää tietää suunniteltaviin rakenteisiin mahdollisesti kohdistuvat muutostarpeet. Tämä tarkoittaa tyypillisesti algoritmien lähtötietoina toimivien avoimien parametrien määrittämistä. Jos suunniteltaviin rakenteisiin toteutettavaa muutosta ei ole huomioitu algoritmeja luotaessa, saattaa algo-

ritmien muokkaaminen olla työlästä. Kun algoritmien muutosavaruus on määritetty riittäväksi, toimivat algoritmit hyvin kaikissa tilanteissa. Hyvin rakennettuja algoritmeja tai niiden osia voidaan tarvittaessa hyödyntää myös muissa projekteissa.

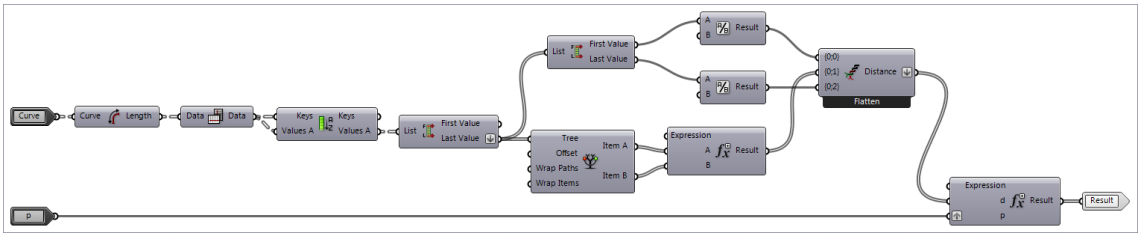
5.1.1 Vahvuudet ja hyödyntämismahdollisuudet

Tässä tutkimuksessa kehitettävä algoritmiavusteinen suunnitteluprosessi toimii kaikilta osiltaan hyvin. Tutkimuksen perusteella algoritmisilla menetelmillä on monia hyödyntämismahdollisuuksia ja niillä on monia etuja perinteisiin menetelmiin verrattuna. Algoritmiavusteisen suunnittelun suurimmat vahvuudet ovat erityisesti geometrisesti vaativien sekä toistuvien rakenteiden suunnittelussa. Tässä tutkimuksessa hyödynnettävän kattorakenteen muotoa voidaan pitää hyvänä esimerkkinä haastavasta geometriasta. Vaikka kantava rakenne muodostuukin pääasiassa suorista rakenneosista, on kaarevan kokonaismuodon luominen hyvin haastavaa perinteisin menetelmin. Lisäksi kattorakenne koostuu toistuvista rakenneosista, joiden hallitsemiseen ja muokkaamiseen algoritmit soveltuvat hyvin. Koska kattorakenteen viivageometria luodaan jakamalla kattopinta kolmionmuotoisiin pintoihin kahden avoimen numeroparametrin avulla, voidaan rakenneosien tiheysjakoa muokata helposti näitä kahta parametria muuttamalla. Näin ollen katon kaarevan muodon mukaan luotavien rakenneosien jakoa voidaan muuttaa algoritmien avulla reaaliaikaisesti, kun perinteisin menetelmin samojen muutosten tekeminen olisi hyvin työlästä ja aikaa vievää.

Haastavien muotojen ja toistuvien rakenteiden lisäksi algoritmisia menetelmiä hyödynnetään erityisesti suunnittelutiedon linkittämisessä. Perinteisessä suunnitteluprosessissa erilaisten suunnittelutietoa sisältävien ohjelmien ja tiedostojen välinen tiedonsiirto on hyvin manuaalista ja työlästä. Kun kaikki suunnittelutieto luodaan algoritmien avulla, voidaan suunnittelutiedon luovat algoritmit linkittää toisiinsa ja näin vältytään mahdollisilta tiedonsiirto-ongelmilta. Tällöin kaikki perinteiset suunnitteluohjelmat, jotka voidaan linkittää algoritmeihin, voidaan linkittää myös toisiinsa. Tässä tutkimuksessa käytettäviä suunnitteluohjelmia ovat Tekla Structures, RFEM, Mathcad ja Excel. Suunnittelutiedon integraatio vähentää päällekkäistä työtä ja ristiriitaisuuksien syntymistä. Koska kaikkia suunnitteluohjelmia käytetään samojen rakenteiden suunnitteluun, on niissä paljon samaa suunnitteludataa. Algoritmien avulla riittää, että tämä suunnittelutieto luodaan kerran ja sen jälkeen se voidaan linkittää helposti kaikkiin tarvittaviin malleihin. Kun esimerkiksi kattorakenteen viivageometria on muodostettu, voidaan sitä hyödyntää sekä laskenta- että rakennemallin luomisessa. Kun geometriatieto muuttuu, päivittyvät muutokset automaattisesti molempiin malleihin. Tällöin suunnitelmat pysyvät ajantasaisina sekä ristiriidattomina keskenään.

Algoritmisen suunnittelun avulla voidaan myös parantaa suunnittelun tarkkuutta. Tyypillisesti suunnittelussa ja erityisesti rakenteiden mitoittamisessa tehdään aina jonkinasteisia yksinkertaistuksia. Tämä tehdään pääasiassa siksi, että suunnittelusta tulisi helpompaa ja nopeampaa. Esimerkiksi toistuvia samankaltaisia kuormia määritettäessä saatetaan työn

nopeuttamiseksi tehdä yksinkertaistus, jossa kuormista valitaan määräävin ja sitä käytetään kaikissa rakenteissa. Kun laskentamallin luomisessa ja kuormien määrityksessä hyödynnetään algoritmeja, voidaan myös tarkat kuormitukset luoda nopeasti. Kun algoritmi luo rakenneosiin kohdistuvat viivakuormat automaattisesti kuormitusalojen mukaan, syntyvät tarkat kuormitustiedot yhtä nopeasti kuin yksinkertaistetutkin. Todellisten kuormien generoimiseen käytettävä algoritmi on tässä tapauksessa hyvin yksinkertainen ja siten sen luominen on käy myös nopeasti. Kuvassa 48 esitetään algoritmi, joka muodostaa kattorakenteen sauvoihin kohdistuvat viivakuormat vallitsevan pinta-alakuorman ja kuormituspituuksien avulla.



Kuva 48. *Rakenteisiin kohdistuvat viivakuormat määrittävä algoritmi.*

Suunnittelun nopeutumisen ja tehostumisen lisäksi algoritmisilla suunnittelumenetelmillä voidaan saavuttaa suuria kustannussäästöjä. Tämä perustuu siihen, että algoritmit ovat hyödyllinen työkalu erityisesti hankkeen alkuvaiheessa tapahtuvaan alustavaan suunnitteluun. Koska rakennushankkeen kokonaiskustannukset määräytyvät suurelta osin aivan hankkeen alkuvaiheessa, voidaan luonnossuunnittelussa tehtävillä ratkaisuilla vaikuttaa merkittävästi hankkeen kokonaiskustannuksiin. Algoritmien luoma automaatio ja niiden parametrinen luonne mahdollistavat erilaisten vaihtoehtojen nopean vertailun. Suunnittelun alkuvaiheessa ei myöskään tarvitse vielä kiinnittää huomiota pienimpiin yksityiskohtiin, jolloin rakenteet luovien algoritmien logiikan määrittäminen on helpompaa. Algoritmien avulla voidaan luoda perinteisiä menetelmiä nopeammin hyvin erilaisia rakenteita ja niiden muodostamia kokonaisuuksia. Kun geometriatieto voidaan vielä helposti linkittää laskentaohjelmaan, päästään tarkastelemaan nopeasti erilaisten rakennejärjestelmien toimivuutta. Kun rakenteisiin linkitetään vielä kustannustietoa, voidaan eri rakennejärjestelmien kustannuksia vertailla nopeasti.

5.1.2 Ongelmat ja puutteet

Vaikka tässä työssä toteutettava algoritminen suunnitteluprosessi toimii kaikilta osiltaan hyvin, törmätään tutkimusta tehdessä myös moniin ongelmiin ja puutteisiin. Suurin koko prosessia koskeva ongelma on algoritmisen tiedonsiirron vaikeus. Jotta suunnittelukokonaisuus pysyisi selkeänä ja jotta siinä käytettäviä algoritmeja voidaan hyödyntää myös tulevaisuudessa, tulee suunnittelukokonaisuutta jakaa erillisiin algoritmimalleihin. Tämä tarkoittaa käytännössä erillisten Grasshopper-tiedostojen käyttöä. Lisäksi kun suunnitteluun osallistuu useita eri suunnitteluosapuolia, on erillisten tiedostojen käyttäminen suo-

tavaa. Algoritmeja voidaan helposti kopioida tiedostosta toiseen, mutta tällöin niiden välinen älykäs linkitys katoaa. Algoritmimallien väliseen tiedonsiirtoon on erilaisia tapoja ja lisäosia, mutta monet niistä koetaan prosessin kannalta huonoiksi.

Vaikka algoritmiavusteisen suunnittelun vahvuus on yksinkertaisessa ja tiiviissä muodossa säilyvä suunnittelutieto, saattaa suuri tietomäärä silti muodostua ongelmaksi. Algoritmimallit ovat tiedostokooltaan huomattavasti perinteisiä suunnittelumalleja pienempiä, mutta erityisesti tiedonsiirto ja suunnittelumallien generoiminen vaativat silti paljon laskentatehoa. Algoritmit kannattaa jakaa prosessin selkeyttämiseksi erillisiin algoritmimalleihin, jotka sitten voidaan linkittää toisiinsa esimerkiksi Speckle-lisäosan avulla. Tutkimusta tehdessä huomataan, että Speckle-komponenttien kautta tapahtuvassa tiedonsiirrossa saavutetaan melko nopeasti tiedonsiirtomaksimi. Tämä tarkoittaa sitä, että tietoa lähettävä Speckle-komponentti ei pysty lähettämään kovinkaan suuria tietomääriä. Grasshopper-tiedostoon voidaan asettaa samanaikaisesti useita tietoa lähettäviä komponentteja, minkä avulla tämä ongelma voidaan ratkaista. Tämä kuitenkin vähentää suunnittelun selkeyttä huomattavasti. Speckle on vielä suhteellisen uusi lisäosa, joten tällaisiin ongelmiin saattaa tulla parannuksia tulevaisuudessa.

Algoritmiavusteisessa suunnittelussa perinteiset suunnittelumallit, kuten esimerkiksi laskenta- ja rakennemalli luodaan algoritmien avulla. Tutkimuksessa tullaan siihen johtopäätökseen, että suunnittelussa on kiinnitettävä erityistä huomiota algoritmiavusteisten ja perinteisten menetelmien rajapintaan. Jos esimerkiksi algoritmien avulla luotava tietomalli tallennetaan ja algoritmeja muokataan tämän jälkeen, generoivat algoritmit uudet osat vanhojen päälle. Tällöin tietomalli sisältää kaksinkertaiset, päällekkäiset rakenneosat, mikä on tietysti väärin. Tästä syystä aina kun suunnittelua jatketaan suunnitteluohjelmiston sulkemisen jälkeen, avataan tyhjä tiedosto, mihin rakenteet generoidaan uudestaan algoritmien avulla. Kattorakenteen kohdalla laskenta- ja rakennemallin tietomäärät säilyvät maltillisina ja näin ollen ne rakentuvat tyhjään tiedostoon nopeasti. Jos tietomallien koko kuitenkin kasvaa huomattavasti suuremmaksi, saattaa mallin generoimiseen kuluva aika olla huomattavasti suurempi. Erityisesti Teklaan luotavan rakennemallin kohdalla tämä saattaa aiheuttaa ongelmia. Tällöin rakennemalli kannattaa luoda useiden algoritmimallien avulla ja tallentaa tietomalli normaaliin tapaan. Kun rakennemallia muokataan, tulee mallista poistaa muokattavat rakenneosat ja luoda ne uudestaan algoritmien avulla. Jos muokattavat rakenteet vaikuttavat rakennemallin muihin osiin, saattaa rakennemalli rakentua väärin ja luoda malliin ristiriitaisuuksia.

Rakennusten suunnittelussa pyritään käyttämään mahdollisimman paljon tyyppiratkaisuja, jotta rakenteet ovat helposti toteutettavissa ja jotta niiden kustannukset säilyvät maltillisina. Rakennukset voidaan kuitenkin muodostaa hyvin monenlaisista rakenneosista ja liitoksista, minkä vuoksi ne eroavat aina jonkin verran toisistaan. Tämä aiheuttaa haasteita algoritmiavusteiselle suunnitteluprosessille. Algoritmien avulla luodaan logiikka siitä, miten erilaiset suunnitteluratkaisut muodostuvat. Jotta algoritmit toimisivat hyvin,

tulee niitä luotaessa ottaa huomioon kaikki mahdolliset rakenteisiin kohdistuvat muutokset. Tämä on erityisesti suurien suunnittelukokonaisuuksien kohdalla käytännössä mahdotonta. Algoritmien avulla voidaan luoda esimerkiksi Tekla-malliin kaikki tarvittavat rakenneosat ja liitokset, mutta niiden muodostaminen saattaa olla huomattavasti hitaampaa perinteisiin menetelmiin verrattuna. Jos suunniteltavat rakenteet muodostavat esimerkiksi monimutkaisen kokonaisuuden ilman matemaattisia säännönmukaisuuksia, on hyvin vaikeaa luoda algoritmeja, jotka pystyvät reagoimaan rakennejärjestelmään kohdistuviin muutoksiin. Tällöin algoritmiavusteisella suunnittelulla ei saavuteta minkäänlaista lisäarvoa perinteisiin menetelmiin verrattuna. Algoritmien avulla voidaan luoda myös logiikkaa, joka pystyy reagoimaan monimutkaisiin kokonaisuuksiin. Tämä on kuitenkin hyvin työlästä, ja jos algoritmeja ei pystytä enää tulevaisuudessa hyödyntämään, voidaan niiden käyttöä pitää tässä tapauksessa turhana.

5.2 Algoritmiavusteisten suunnittelutyökalujen toimivuus

Algoritmisen suunnitteluprosessin toimivuus perustuu käytännössä prosessissa käytettävien työkalujen toimivuuteen. Työkaluina toimivat Grasshopperin komponentit ja lisäosat sekä niiden avulla muodostetut algoritmit. Jotta algoritmeja voidaan hyödyntää mahdollisimman tehokkaasti, tulee ne ryhmitellä ja järjestellä selkeiksi kokonaisuuksiksi. Lisäksi algoritmit kannattaa jäsennelle erillisiin algoritmimalleihin prosessin selkeyttämiseksi. Rakennesuunnittelun kannalta algoritmiset työkalut voidaan jakaa karkeasti kahteen perustyyppiin: rakenteiden mitoittamisessa ja mallintamisessa käytettäviin työkaluihin. Suunnitteluprosessin kannalta tärkeitä ovat myös tiedonsiirtoon käytettävät työkalut. Algoritmien käytettävyyden parantamiseksi algoritmit ja niiden osat kannattaa ryhmitellä, jäsennellä, nimetä ja numeroida selkeästi. Tähän voidaan hyödyntää sekä valmiita työkaluja että itse luotuja algoritmeja. Vaikka algoritmien jäsentely saattaa hidastaa suunnittelun etenemistä jonkin verran, koetaan sen olevan hyvin tärkeää algoritmien käytettävyyden ja jatkokehityksen kannalta.

Grasshopperiin on tarjolla monia erilaisia lisäosia, jotka tekevät suunnittelusta helpompaa ja tehokkaampaa. Pääasiassa Grasshopperiin kehitetyt komponentit ja lisäosat toimivat hyvin. Koska monet lisäosat ovat vielä hyvin uusia ja kehitysvaiheessa, lisääntyvät ja parantuvat niiden ominaisuudet ajan kuluessa. Saatavilla olevia lisäosia on jo useita satoja ja niiden määrä kasvaa jatkuvasti. Tästä syystä suunnittelijat eivät välttämättä edes tiedä kaikkien suunnittelua helpottavien työkalujen olemassaolosta. Jos johonkin tehtävään ei ole saatavilla tarkoituksenmukaista komponenttia, voi sellaisen luoda helposti myös itse. Tähän voi käyttää joko visuaalista tai tekstimuotoista ohjelmointia. Algoritmin osakokonaisuuksista voidaankin muodostaa sisäisiä ryhmiä eli niin sanottuja clustereita. Tällöin algoritmikokonaisuuden analysointi helpottuu ja nopeutuu.

5.2.1 Tiedonsiirto

Tiedonsiirtoon kokeillaan kahta erilaista menetelmää, joista ensimmäisessä tiedonsiirto tapahtuu Rhino-tiedostojen kautta. Tässä hyödynnetään erityisesti Elefront-lisäosaa, jonka avulla geometriatietoon voidaan liittää erilaisia ominaisuustietoja ja se voidaan muuntaa pysyväksi Rhinossa. Tiedonsiirron onnistumiseksi geometriatieto ja siihen liitettävä ominaisuustieto joudutaan tallentamaan Rhino-tiedostoon. Kun suunnitteluprosessi koostuu useista algoritmimalleista, vaaditaan tiedonsiirron onnistumiseksi melko paljon manuaalista työtä. Erityisesti muutosten tekeminen koetaan työlääksi, koska algoritmit eivät päivity automaattisesti eivätkä komponentit pysty varastoimaan tietoa muuhun kuin Rhino-tiedostoon. Näin ollen prosessin eri vaiheet tulee tallentaa aina omaan Rhino-tiedostoonsa tai vähintään muuntaa tieto pysyväksi Rhinossa. Tiedonsiirto perustuu algoritmien luoman suunnittelutiedon siirtämisen Rhinoon manuaalisesti, minkä vuoksi tiedonsiirto koetaan algoritmiavusteisen prosessin kannalta puutteelliseksi. Lisäksi kaikki tieto pitää linkittää geometriatietoon, mikä ei ole useinkaan tarpeellista. Elefrontin avulla ei myöskään pysty siirtämään muuta tietoa kuin geometriaa ja yksinkertaista teksti- ja numerodataa.

Toinen algoritmimallien linkitystapa on pilvipohjainen tiedonsiirto. Tutkimuksen alussa markkinoilla on saatavilla kaksi erilaista tähän soveltuvaa Grasshopper-lisäosaa: Flux ja Konstru. Molemmat näistä ohjelmista sisältävät tiedon lähettämiseen ja vastaanottamiseen käytettäviä Grasshopper-komponentteja sekä lisäosia erilaisiin suunnittelussa käytettäviin suunnitteluohjelmistoihin. Konstrua käytettäessä käytännössä kaikki suunnittelutieto tulee luokitella Konstrun omien komponenttien avulla. Esimerkiksi yksinkertaista geometriadataa ei voi siirtää sellaisenaan vaan viivat pitää luokitella Konstrun komponenteilla esimerkiksi teräspalkeiksi- ja pilareiksi. Tämän koetaan olevan suunnittelun etenemisen kannalta turhaa ja työlästä, minkä vuoksi Konstrua ei käytetä tutkimuksen toteutuksessa tätä kokeilua enempää. Flux toimii samalla periaatteella kuin Konstru, mutta se sallii vapaasti kaikenlaisen suunnittelutiedon jakamisen ilman erillisiä luokituksia. Fluxin koetaan olevan hyödyllinen lisäosa algoritmiavusteisessa suunnitteluprosessissa tapahtuvaan tiedonsiirtoon. Tutkimuksen aikana Flux-lisäosa ja sen tarjoama pilvipalvelu lopettavat kuitenkin toimintansa, minkä vuoksi sen käyttäminen on mahdotonta. Tästä syystä tiedonsiirrossa joudutaan turvautumaan muihin ratkaisuihin.

Algoritmimallien välisessä tiedonsiirrossa päädytään käyttämään lopulta Speckle-lisäosaa. Speckle toimii varhaisesta kehitysvaiheestaan huolimatta erittäin hyvin, mutta myös joitakin puutteita löytyy. Specklen toimintaperiaate on hyvin yksinkertainen ja se koetaan lisäosan yhdeksi vahvuudeksi. Speckle ei vaadi erityisiä suunnittelutiedon luokituksia vaan sen kautta voi siirtää tietoa helposti sellaisenaan. Speckleä käytettäessä tarvitaan tyypillisesti ainoastaan kahta komponenttia: tiedon lähettäjää sekä tiedon vastaanottajaa. Tietoa lähettävään komponenttiin voidaan luoda useita syöttöportteja ja ne päivittyvät automaattisesti myös tiedon vastaanottaviin komponentteihin. Tämä mahdollistaa joustavan suunnittelun, jossa useat suunnittelijat voivat esimerkiksi muokata samanaikaisesti

toisiinsa liittyviä rakenteita. Vaikka Specklen kautta on mahdollista siirtää monenlaista tietoa, jotkin vaativammat tietorakenteet hajoavat siirron aikana. Esimerkiksi Teklan sisältämiä mallinnusobjekteja ei voi siirtää Specklen kautta ilman, että tietorakenne hajoaa. Tämä vaikeuttaa rakennemallin luomiseen käytettävien algoritmimallien välistä tiedonsiirtoa huomattavasti.

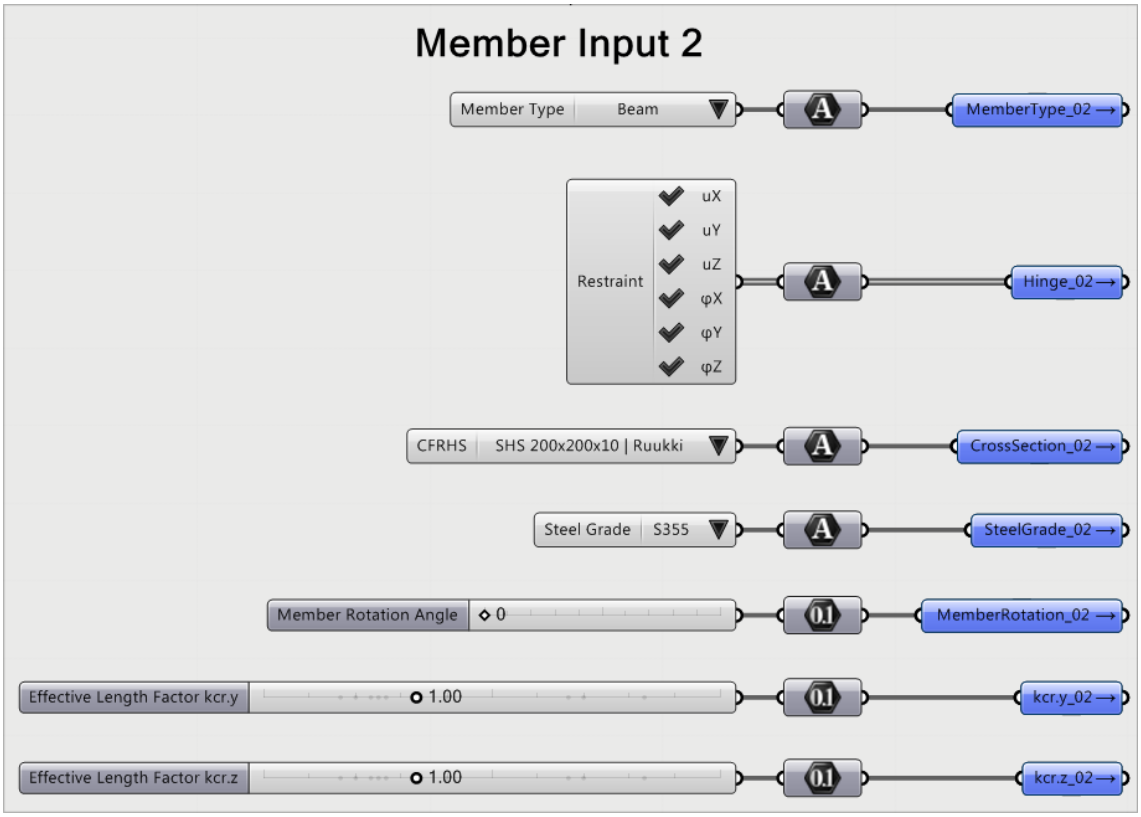
Speckle sisältää myös työkalun, jonka avulla on mahdollista tarkastella kaikkia käytettyjä tiedonsiirtotapauksia ja tarkastella erityisesti komponenttien välillä siirtyvää geometriaa. Geometrian havainnollistaminen ja tarkastelu ei tällä hetkellä toimi, mutta tämän tutkimuksen kannalta sillä ei ole suurta merkitystä. Lisäksi tutkimuksessa koetaan ongelmalliseksi se, että lähtökohtaisesti Speckle luo kaikesta sen kautta siirrettävästä tiedosta julkista. Jos suunnittelutiedosta haluaa tehdä yksityistä, joutuu suunnittelija käyttämään Specklen käyttöliittymää Rhinon kautta ja merkitsemään yksitellen kaikki tiedonsiirtodata yksityiseksi. Jokaisessa tiedonsiirtotapauksessa Speckle generoi lähetettävälle tiedolle automaattisesti yksilöllisen tunnuksen. Tätä tunnusta voidaan tämän jälkeen käyttää tiedon vastaanottamiseen. Lähetystunnusta ei voi määrittää itse, minkä johdosta saman tunnuksen alle ei voi määrittää useita tiedonsiirtotapauksia. Tästä syystä tiedon vastaanottajalle tulee aina ilmoittaa lähetystunnus, jonka avulla hän pääsee hyödyntämään lähettäjän luomaa suunnittelutietoa.

5.2.2 Rakenteiden algoritmiavusteinen mitoitus

Tässä tutkimuksessa rakenteiden mitoittamisessa hyödynnetään Mathcad- ja RFEM-laskentaohjelmia. Ne ovat rakennesuunnittelussa laajasti käytettyjä ohjelmistoja, minkä ansiosta niiden toimintaperiaate on monille suunnittelijoille valmiiksi tuttu. Molempia laskentaohjelmia ohjataan niille kehitetyn lisäosan avulla. RFEM on ominaisuuksiltaan hyvin monipuolinen ohjelma ja siksi siinä on myös monia algoritmien avulla ohjattavia osia. RFEM-laskentamallin luomisessa ja muokkaamisessa käytetään A-Insinöörit Suunnittelu Oy:n itsekehittämää GH-RFEMLink-lisäosaa. Lisäosa on suhteellisen uusi ja sen kehitystyö jatkuu koko tutkimuksen ajan. Tästä syystä lisäosan ominaisuudet lisääntyvät ja parantuvat useita kertoja tutkimuksen aikana. Mathcad-ohjelmaa ohjataan Macaw-lisäosan avulla. Koska Mathcad on huomattavasti yksinkertaisempi ohjelma kuin RFEM, on myös siihen linkitettävien algoritmien määrittäminen helpompaa.

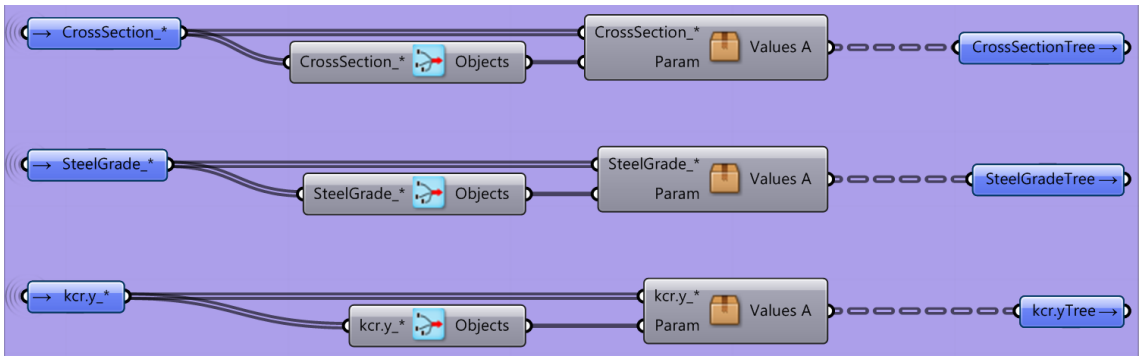
Vaikka RFEM-laskentamallia ohjaavien algoritmien määrittämien on melko haastavaa, voidaan algoritmeja käyttää melko helposti myös tulevaisuudessa. Tutkimuksessa pyritäänkin luomaan algoritmeja ja niiden muodostamia kokonaisuuksia siten, että niitä voidaan hyödyntää helposti myös muissa projekteissa. GH-RFEMLinkin avulla tapahtuva laskentamallin luominen ja muokkaaminen toimivat pääasiassa moitteita. Lisäosan avulla pystytään tällä hetkellä määrittämään suhteellisen hyvin kaikki teräsrakenteiden FEM-laskennassa tarvittavat ominaisuudet. Mallin luominen tapahtuu tyypillisesti kolmessa vaiheessa. Ensimmäiseksi määritetään manuaalisesti syötettävät lähtötiedot, toisessa vai-

heessa tieto muokataan ja järjestetään haluttuun muotoon ja lopuksi ominaisuudet luodaan laskentamalliin. Näistä selvästi haastavin ja työläin osa on toinen vaihe, jossa tieto muokataan haluttuun muotoon. Lähtötietojen määrittäminen ja laskentamallin luominen tapahtuvat aina samalla tavalla projektista riippumatta, mutta oikeanlaisen tietomuodon rakentaminen täytyy tehdä aina projekti- ja tehtäväkohtaisesti uudestaan. Kuvassa 49 esitetään esimerkki laskentamalliin luotavien sauvojen lähtötietojen määrittelystä.



Kuva 49. *Laskentamallin terässauvojen ominaisuuksien määrittelyä.*

Laskentamallin luomista helpottaa kuvan 49 oikeassa laidassa esiintyvät siniset komponentit, jotka kuuluvat Telepathy-lisäosaan. Kun Telepathy-komponentit nimetään ja numeroidaan järkevästi, voidaan niitä hyödyntää osakokonaisuuksien välisten loogisten yhteyksien rakentamisessa. Tietoa vastaanottavissa komponenteissa on ominaisuus, jonka avulla se voi vastaanottaa useista lähteistä saapuvaa tietoa. Käytännössä tämä tapahtuu yksinkertaisesti asettamalla vastaanottajan nimen perään *-merkki, jolloin se vastaanottaa tiedot kaikista samalla alkutunnuksella varustetuista lähettäjäkomponenteista. Loogisesti toimivan algoritmin kannalta ongelmana on kuitenkin se, mihin järjestykseen vastaanottaja asettaa useista lähteistä saapuvat tiedot. Tämä ongelma voidaan ratkaista nimen perässä käytettävän numerotunnuksen ja Metahopper-komponenttien avulla. Kuvassa 50 esitetään yksinkertainen esimerkki siitä, miten manuaalisesti määritetyt ominaisuustiedot järjestetään automatisoidusti rakenneosat luovan komponentin ymmärtämään muotoon.



Kuva 50. *Rakenneosien ominaisuustietojen muokkaus.*

Laskentamalli voidaan luoda joko vaiheittain erilliset osat generoivien komponenttien tai koko mallin luovan komponentin avulla. Tämä koetaan hyväksi tavaksi toimia, koska tällöin koko mallia ei tarvitse generoida uudestaan yhden ominaisuustiedon muuttuessa. Tämä säästää huomattavasti laskentamallin muokkaamiseen käytettävää aikaa. Laskentamallin ominaisuuksista kuormat vaativat selvästi eniten laskentatehoa ja niiden generoiminen kuluttaakin suurimman osan mallin generoimiseen kuluvasta ajasta. Kattorakenteen kohdalla koko mallin luominen kestää kuitenkin vain muutamia minuutteja.

Kun laskentamalli on luotu ja laskenta suoritettu, on algoritmimallin käyttö suhteellisen suoraviivaista ja helppoa. Ensin suoritetaan teräsrakenteiden Eurokoodin mukainen mitoitus. Mitoituksen suorittavan komponentin käyttö on helppoa. Jos rakenteita ei optimoida, komponentti suorittaa mitoituksen nopeasti. Jos optimointi kytketään päälle, optimoidaan puumuodon jokaisella oksalla sijaitsevaa joukkoa niille asetettujen käyttöasterajojen mukaan. Optimointi toimii siten, että algoritmit suorittavat laskennan ja muuttavat tämän jälkeen profiilitietoja yhden koon suuremmaksi tai pienemmäksi vallitsevan käyttöasteen mukaan. Tämän jälkeen algoritmit suorittavat laskennan uudestaan ja muuttavat taas profiilitietoja tarkoituksenmukaiseen suuntaan. Algoritmit toteuttavat tätä kierrosta automaattisesti, kunnes rakenneosille ovat löytyneet optimoidut poikkileikkaukset. Tämä prosessi saattaa olla tietyissä tapauksissa melko hidasta ja sen koetaan olevan huomattavasti hitaampaa kuin RFEM:n oman optimointityökalun toiminta. Jos profiilitietojen alkuarvaus on lähellä oikeaa ei optimoinnin hitaus aiheuta kuitenkaan suurta ongelmaa.

Mitoituksen jälkeen laskentamallista saatavat tulokset voidaan palauttaa takaisin algoritmimalliin. Tulostieto voidaan tämän jälkeen varastoida esimerkiksi Excel-taulukkoon tai siirtää Specklen avulla toiseen algoritmimalliin. Kuten edellä esitetään, Specklen kohdalla törmätään nopeasti liian suureen datamäärään. Tämä vaikeuttaa prosessin toimintaa, mutta ei tee siitä kuitenkaan mahdotonta. Tulostieto koostuu käytännössä ainoastaan yksinkertaisista numero- ja tekstilistoista, minkä vuoksi tiedonsiirtomaksimin saavuttaminen koetaan hämmentäväksi. Tietoa yritetään pakata eri tavoin ennen kuin se lähetetään eteenpäin Specklen avulla, mutta tähän ongelmaan ei löydetä hyvää ratkaisua tämän tutkimuksen puitteissa.

Toinen mitoituksessa käytettävä ohjelma on Mathcad. Mathcad-laskentapohja koostuu yksinkertaistetusti lähtötietoparametreista, laskentaa ohjaavista matemaattisista kaavoista ja laskennan tuloksista. Mathcadin avulla tapahtuva laskenta on yksinkertaista, ja se pystytään usein toteuttamaan tehokkaammin muilla tavoilla. Rakennesuunnittelussa suoritettava laskenta ja mitoitus pitää kuitenkin tyypillisesti raportoida selkeästi. Tästä syystä Mathcad on alalla yleisesti käytetty ohjelma. Tässä tutkimuksessa Mathcad-laskentapohjaan syötetään algoritmin avulla erilaisia kuormatietoja, jotta voidaan varmistua mitoitettavan liitoksen kestävyydestä eri kuormitustapauksissa. Liitoksen mitoituksessa käytettävä laskentapohja on melko monimutkainen, minkä vuoksi yhden laskentatapauksen suorittaminen kestää jopa hieman yli sekunnin. Tämä on Mathcadin kohdalla suhteellisen pitkä aika. Hoopsnake-lisäosan avulla tapahtuvassa kuormien syötössä nähdään kuitenkin saavutettavat tulokset reaaliaikaisesti ja laskenta voidaan keskeyttää milloin tahansa. Algoritmien avulla ohjattava Mathcad-laskenta koetaankin hyvin potentiaaliseksi tavaksi suorittaa laskentaa.

5.2.3 Rakenteiden algoritmiavusteinen tietomallinnus

Rakennesuunnittelussa yleisesti hyödynnettävä tietomallinnusohjelma Tekla Structures sisältää paljon erilaisia tietomallin luomiseen ja muokkaamiseen käytettäviä työkaluja. Teklaa on mahdollista ohjata algoritmiavusteisesti Tekla Live Linkin avulla ja se mahdollistaa lähes kaikkien Teklan sisältävien ominaisuuksien määrittämisen ja ohjaamisen. Rakennemallin algoritmiavusteisessa suunnittelussa voidaan siis suorittaa käytännössä kaikki samat tehtävät kuin perinteisessäkin tietomallintamisessa. Yleisesti ottaen rakennemallin luominen koetaan huomattavasti haastavammaksi kuin laskentamallin. Tämä johtuu siitä, että laskentamallin kohdalla voidaan tehdä monia yksinkertaistuksia ja siksi se pysyy helpommin hallittavana kokonaisuutena. Rakennemallissa tulee huomioida myös rakenneosien tilavuus, mikä tekee sen luomisesta haastavampaa. Lisäksi rakennemallissa osien tulee olla täsmällisesti juuri oikeassa sijainnissa ja asennossa, kun taas laskentamallissa voidaan tehdä melko karkeitakin yksinkertaistuksia.

Rakennemallin algoritmiavusteinen luominen sisältää myös liitosten mallintamisen. Rakennemallia ohjaavat algoritmit saattavatkin muodostua rakenneosia ja liitoksia generoidessa hyvin monimutkaisiksi. Algoritmiavusteisen tietomallintamisen koetaan sopivan erityisesti laskentamallista saatavien runkorakenteiden mallintamiseen ja niiden muokkaamiseen. Lisäksi algoritmeja voidaan hyödyntää erityisesti pienempien, helpommin havainnollistettavien kokonaisuuksien mallintamisessa. Koska laskentamallin geometria on yksinkertaistus todellisesta, joudutaan sitä muokkaamaan melko paljon rakennemalliin sopivaksi. Jos tässä halutaan ottaa vielä huomioon kaikki mahdolliset rakenteissa tapahtuvat muutokset, algoritmien luominen vaikeutuu. Tästä syystä rakenteiden tietomallinnus on tyypillisesti vain osittain algoritmiavusteista.

Eräs algoritmisessa tietomallintamisessa kohdattava kysymys on, mitkä tekijät kannattaa toteuttaa algoritmiavusteisesti ja mitkä perinteisin menetelmin. Jos osa rakennemallista

luodaan algoritmiavusteisesti ja osa perinteisin menetelmin, tulee suunnittelijan määritellä näiden kahden menetelmän välinen rajapinta. Koska käytännössä kaikki mallintamisen osat voidaan toteuttaa myös algoritmiavusteisesti, on hyvin vaikeaa sanoa yksiselitteisesti, mihin tehtäviin algoritmit soveltuvat perinteisiä menetelmiä paremmin. Tässä tutkimuksessa ehditään kokeilemaan vain murto-osaa Teklan ominaisuuksista ja niidenkin toteutustapa saattaa olla erilainen kokonaisuudesta riippuen. Tutkimuksessa kokeillaan esimerkiksi sauvojen päiden sijainnin parametrista muuttamista. Tässä kokeillaan ensin Tekla-objektit luovan komponentin attribuuttitietojen muokkaamista. Päätepisteiden epäselvän ja vaikeasti havainnollistettavan koordinaatiston vuoksi tämä koetaan hyvin vaikeaksi. Tästä syystä tässä esimerkkitapauksessa päädytään muokkaamaan algoritmeilla suoraan alkuperäistä viivageometriaa ja näin saavutetaan haluttu lopputulos.

Yksi oleellinen osa algoritmiavusteista tietomallinnusta on liitosten mallintaminen. Erityisesti algoritmisessa prosessissa siihen voidaan käyttää hyvin moninaisia tapoja. Tyypillisesti liitosten mallinnuksessa pyritään käyttämään mahdollisimman paljon valmiita liitoskomponentteja. Jos liitoksen luova komponentti toimii hyvin, voidaan sillä luoda liitos hyvin joustavasti erilaisiin paikkoihin. Joskus komponentit eivät kuitenkaan taivu haluttuun muotoon ja mallinnuksessa joudutaan turvautumaan esimerkiksi komponentin räjäyttämiseen. Algoritmisessa prosessissa liitosten mallinnusta voidaan lähestyä hieman eri tavalla. Koska algoritmien avulla voidaan luoda melko helposti liitososien välisiä loogisia yhteyksiä, voidaan liitokset luoda yksittäisiä osia mallintamalla. Työssä toteutetun jatkosliitoksen kohdalla kokeiltiin sekä valmista liitoskomponenttia että erillisistä liitososista muodostettavaa liitosta. Liitoskomponenteissa saattaa olla hyvin monia asetuksia, joita suunnittelija voi määrittää. Algoritmisessa mallinuksessa on melko työlästä määrittää liitoksen ominaisuudet määrittävät attribuuttitiedot oikein. Tästä syystä erillisten liitososien yksittäinen mallintaminen saattaakin olla valmiiden liitoskomponenttien käyttöä helpompaa. Kuvassa 51 esitetään esimerkki päätylevyliitoksen ominaisuudet määrittävistä attribuuteista. Attribuuttitiedot eivät lopulta ole kovinkaan monimutkaiset, mutta niiden tarkoituksenmukainen määrittäminen koetaan melko työlääksi.

	{0}
0	e1 125;e2 125;tpl1 20;bpl1 330;hpl1 540;diameter 20;tolerance 2;nb 2;lbd " 430 ";rb1 55;nw 3;lwd " 100 ";rw1 65;rw2 65;prefix_pos1 "PL";startno_pos1 1;cut 1;mat "S355J2";screwdin "EN 15048-1";thread_in 0

Kuva 51. *Päätylevyliitoksen attribuuttitiedot.*

Kun erilliset liitososat mallinnetaan erikseen, säilyy tietomallin sisältö joustavana. Tämä tarkoittaa sitä, että yksittäisten levyjen, pulttiryhmien ja hitsiliitosten avulla voidaan muodostaa hyvin monimuotoisia liitoksia. Koska valmiit liitoskomponentit sisältävät aina jonkin verran käyttäjän saavuttamattomissa olevaa logiikkaa, eivät ne sovellu kaikkiin

mahdollisiin tilanteisiin. Valmiita liitoskomponentteja käytetään perinteisessä suunnittelussa paljon, koska niiden käytettävyys on normaalissa tapauksessa huomattavasti nopeampaa. Algoritmiavusteisessa suunnittelussa näin ei kuitenkaan aina ole. Kun suunnittelija luo itse algoritmien avulla liitososien välisen logiikan, on normaalisti hyvin manuaalinen mallintaminen nopeaa. Tällöin suunnittelija voi myös luoda suhteellisen helposti liitoksia, joihin valmiit liitoskomponentit eivät sovellu. Koska erilaisia liitostyyppejä on todella paljon, on mahdotonta sanoa yksiselitteistä ohjetta, miten liitokset kannattaa mallintaa algoritmiavusteisessa projektissa. Tilannetta tulee tarkastella aina tapauskohtaisesti.

5.3 Jatkotutkimuskohteet

Algoritmiavusteinen suunnittelu on käsitteenä hyvin laaja ja sen vuoksi on vaikeaa selvittää kaikkia siihen kohdistuvia jatkotutkimustarpeita. Koska algoritmiavusteisia menetelmiä hyödynnetään vielä suhteellisen vähän rakennesuunnittelussa, on siinä monia kehitettäviä tekijöitä. Monia algoritmiavusteisten menetelmien mahdollistavia toimintatapoja ei olekaan vielä edes kokeiltu rakennesuunnittelun tehtäväkentässä. Algoritmiavusteisen suunnittelun mahdollistama potentiaali on noussut viime vuosina laajasti myös rakennesuunnittelijoiden tietoisuuteen. Näin ollen monia algoritmiseen suunnitteluun liittyviä tekijöitä kokeillaan ja kehitetään kovaa vauhtia eteen päin.

Tässä tutkimuksessa keskitytään tarkastelemaan erityisesti teräsrakenteiden algoritmiavusteista suunnitteluprosessia. Yksi selkeä jatkokehityskohde on laajentaa algoritminen suunnittelu myös muihin yleisesti käytettyihin materiaaleihin, kuten esimerkiksi betoniin ja puuhun. Tekla Live Link sisältää hyvin kattavat työkalut mallintaa erilaisia betoniosia ja raudoitteita, minkä ansiosta myös betonirakenteiden algoritmiavusteinen suunnittelu on mahdollista. Betonirakenteiden algoritmiavusteinen mitoitus GH-RFEM-Link-lisäosaa hyödyntämällä on myös oleellinen kehityskohde. RFEM-laskentaohjelmasta voidaan tulevaisuudessa saada algoritmisesti tarvittavat raudoitusmäärät, minkä jälkeen betonirakenteet ja niissä sijaitsevat raudoitteet voidaan mallintaa automatisoidusti rakennemalliin. Erilaisten rakenteiden mitoituksessa ja mallinnuksessa on vielä monia osakokonaisuuksia, joiden soveltuvuutta algoritmiavusteiseen suunnitteluun ei ole päästy testaamaan. Moniin tehtäviin on jo olemassa tarvittavat työkalut, ja näin ollen kynnys uusien asioiden kokeilemiseen ja kehittämiseen on alhainen.

Koska algoritmien avulla ohjataan usein jotakin perinteistä suunnitteluohjelmaa, tulee suunnittelijan arvioida, mitä kannattaa toteuttaa algoritmien avulla ja mitä perinteisin menetelmin. Lisäksi tulee tarkastella, kannattaako laskentaprosessi suorittaa algoritmien avulla Grasshopperissa vai suunnitteluohjelmistossa. Laskentaohjelmat kuten RFEM sisältävät monia mallin luomista ja muokkaamista helpottavia työkaluja. Esimerkiksi rakenteisiin kohdistuvia kuormia voidaan määrittää useilla eri tavoilla. Yksi hyödyllinen työkalu on sauvamaisiin rakenteisiin kohdistuvien viivakuormien määrittäminen aluekuormista. Tämä sama asia voidaan toteuttaa kuitenkin myös algoritmien avulla

Grasshopperissa. Tässä tutkimuksessa viivakuormat määritetään Grasshopperissa, mutta tulevaisuudessa tulee miettiä, kumpi tapa on todellisuudessa tehokkaampi. Sama ongelma kohdataan myös monien muiden suunnitteluohjelmistojen ja työkalujen kohdalla. Siksi tätä asiaa on hyvä tutkia vielä lisää.

Tässä tutkimuksessa teräsrakenteiden poikkileikkaukset optimoidaan RFEM-laskentaohjelmaa hyväksi käyttäen. Jatkossa tämä optimointi voidaan mahdollisesti laajentaa myös kokonaisten rakennejärjestelmien optimointiin. Optimoinnin toteutus on tällä hetkellä vielä melko hidasta, mutta sitä on mahdollista kehittää. Kun optimointi kohdistetaan kokonaisiin rakennejärjestelmiin siten, että niiden geometria voi muuttua, tekee se optimoinnista selvästi haastavampaa. Jos kokonaisten rakennejärjestelmien älykäs optimointi saadaan toteutettua hyvin, saatetaan sillä saavuttaa hyvin merkittäviä kustannussäästöjä. Jos rakennejärjestelmiä aletaan optimoida rakennesuunnittelussa niiden geometriaa muokkaamalla, saattaa se vaikuttaa koko suunnitteluprosessin toteutukseen. Koska arkkitehti päättää tyypillisesti päägeometriasta heti suunnitteluprosessin alkuvaiheessa, vaatii kokonaisvaltainen optimointi täysin uudenlaista yhteistyötä. Optimoinnilla voidaan kuitenkin saavuttaa hyvin merkittäviä etuja, minkä vuoksi se on tärkeä jatkotutkimuskohde.

Erityisesti koko suunnitteluprosessin kannalta oleellinen tutkimuskohde on prosessissa tapahtuva tiedonsiirto. Pilvipohjainen tiedonsiirto koetaan tutkimuksessa oivalliseksi menetelmäksi, mutta myös kehitettäviä asioita löytyy. Erityisesti suurten tietomäärien siirtoon ja varastointiin tulee kiinnittää huomiota. Jos suunnittelutietoa ja erityisesti muodostettuja algoritmeja voidaan varastoida jollakin järkevällä tavalla, voidaan niitä käyttää tehokkaasti myös tulevaisuudessa. Jos suunnittelussa käytettävät algoritmit luodaan systemaattisesti ja raportoidaan selkeästi, voidaan niitä hyödyntää muissakin projekteissa. Kaikista algoritmeista voisi olla mahdollista rakentaa laaja datapankki, josta niitä voitaisiin käydä älykkäästi hakemassa samankaltaisen suunnittelutehtävän sattuessa eteen. Vaikka tässä tutkimuksessa rakennesuunnittelija saa lähtötietonsa arkkitehdilta pilvipalvelun kautta, voidaan tätä menetelmää yhä jatkokehittää. Koska algoritmien avulla tapahtuva reaaliaikainen, pilvipohjainen tiedonsiirto koetaan hyväksi menetelmäksi, kannattaa sen mahdollisuuksia kokeilla myös muiden rakennushankkeeseen osallistuvien tahojen välillä.

Eräs oleellinen algoritmiavusteisen suunnittelun jatkotutkimuskohde on suunnittelun automaatioaste. Suunnitteluprosessi voidaan tarvittaessa automatisoida algoritmien avulla hyvin pitkälle, mutta rakennesuunnittelun monimuotoisuuden vuoksi tämä ei ole aina järkevää. Tutkimuksessa jatkosliitoksen suunnittelu toteutetaan siten, että suunnittelija itse valitsee liitoksen lähtötietoparametrit ja tarkistaa liitososien kestävyyden. Jos liitososat eivät kestä, joutuu suunnittelija muuttamaan lähtötietoparametreja manuaalisesti ja tarkistamaan liitoksen kestävyyden uudestaan. Tämä voidaan toteuttaa myös algoritmiavusteisesti siten, että algoritmit muokkaavat liitosparametrejä automaattisesti käyttöasteen

mukaan. Koska suunnittelussa saattaa olla usein tekijöitä, joita on hyvin vaikeaa ohjelmoida yksiselitteisesti algoritmeihin, jätetään tässä tutkimuksessa lähtötietoparametrit avoimeksi. Tämä voidaan kuitenkin toteuttaa myös automatisoidusti, mutta se jää tulevaisuuden haasteeksi.

Visuaalisen ohjelmoinnin avulla tuotetut algoritmit toimivat hyvin tehokkaasti eikä niiden toiminnassa nähdä suuria puutteita. Hyvin monimutkaisten kokonaisuuksien kohdalla ne saattavat kuitenkin muodostua vaikeaselkoisiksi. Tällöin saattaa olla viisaampaa luoda logiikka perinteisen, tekstimuotoisen ohjelmoinnin avulla. Yksi tulevaisuuden jatkotutkimuskohde on tekstimuotoisen ja visuaalisen ohjelmoinnin hyödyt ja haasteet rakennesuunnittelussa. Koska rakennesuunnittelijoilla ei tyypillisesti ole aiempaa kokemusta ohjelmoinnista, visuaalinen ohjelmointi on helppo tapa lähestyä aihetta. Aloittelijan on helpompi ymmärtää ja siten myös muokata visuaalisia ohjelmointialgoritmeja. Tämä ei kuitenkaan ole pitkällä aikavälillä tehokkain ratkaisu. Tekstimuotoinen ohjelmointi saattaa soveltua vielä paremmin rakennesuunnittelun toteutukseen ja tehdä siitä yhä tehokkaampaa, ja siksi se on yksi tärkeä jatkotutkimuskohde.

Algoritmiavusteinen suunnittelu etenee algoritmeihin ohjelmoitujen sääntöjen mukaan. Nykyään suunnittelussa voidaan hyödyntää myös tietokoneen tekoälyä. Tämä onkin hyvin merkittävä jatkotutkimuskohde. Tekoälyä on mahdollista hyödyntää rakennesuunnittelussa hyvin laajasti. Tekoälyä voidaan hyödyntää hyvin eritasoisissa ongelmissa, mutta parhaimmillaan se voi oppia tunnistamaan rakenteisiin kohdistuvat kuormitukset ja luomaan automaattisesti niihin soveltuvat rakenneosat. Esimerkiksi liitossuunnittelussa algoritmit luovat täsmällisesti sellaisen liitoksen kuin suunnittelija määrittää. Kun liitossuunnittelussa hyödynnetään tekoälyä ja koneoppimista, suunnittelee tietokone liitoksen oman mielensä mukaan sille annetun opetusdatan puitteissa. Tämä tarkoittaa sitä, että jos tekoälyn halutaan osata muodostavan tyypillisesti käytettyjä rakenteita ja liitoksia tulee niille syöttää riittävän paljon monipuolista opetusdataa. Tekoälyn hyödyntäminen on hyvin monimutkainen ja haastava kokonaisuus ja se on yksi merkittävimmistä algoritmiavusteisen suunnittelun jatkotutkimuskohteista.

6. YHTEENVETO

Rakennusten suunnitteluun ja toteutukseen osallistuu yhä useampia eri tahoja ja toimijoita. Tämän seurauksena rakennushankkeet muuttuvat yhä monimutkaisemmiksi ja vaikeammin hallittaviksi kokonaisuuksiksi. Tällöin eri osapuolien välisen yhteistyön ja vuorovaikutustaitojen tärkeys korostuu. (Junnonen & Kärnä 2015; Koskenvesa 2010) Rakennesuunnittelijan työn kannalta merkittävin hankeosapuoli on tyypillisesti arkkitehti, koska arkkitehtisuunnitelmat toimivat rakennesuunnittelun lähtötietona. Arkkitehdin ja rakennesuunnittelijan tulee käydä suunnitteluprosessin edetessä jatkuvaa vuoropuhelua, jotta suunnitelmat pysyvät ristiriidattomina ja virheettöminä. Rakennesuunnittelu on monivaiheinen iteratiivinen prosessi, joka etenee alustavista luonnossuunnitelmista toteutettaviksi rakenteiksi ja liitoksiksi. Huomattava aika suunnittelusta menee suunnitelmien muokkaamiseen ja siksi muutostenhallinnan tulee olla mahdollisimman helppoa ja tehokasta.

Nykyään monet rakennesuunnitteluun kuuluvat tehtävät voidaan suorittaa erilaisia tietokoneohjelmia hyväksi käyttäen. Ohjelmistojen käyttö helpottaa ja nopeuttaa suunnittelua, mutta toisaalta ne myös luovat täysin uusia työtehtäviä. Ohjelmistot keskittyvät usein vain johonkin tiettyyn rajattuun aiheeseen, minkä vuoksi suunnittelijan on huolehdittava eri osakokonaisuuksien yhteensovittamisesta. Yksi merkittävä suunnittelussa kohdattava ongelma on erilaisten suunnittelutietojen yhdistämisen ja tiedonsiirron vaikeus. Ongelmana ovat eri hakeosapuolien sekä yksittäisten suunnittelijoiden käyttämät erilaiset ohjelmistot, jotka käyttävät erilaisia tiedostomuotoja. Tällöin tiedonsiirto joudutaan toteuttamaan usein manuaalisesti, mikä on hyvin hidasta ja häiriöherkkää. Tämä johtaa siihen, että tiedonsiirtoa ei tehdä kuin tarvittaessa. Nopeasti etenevässä suunnittelussa tämä on ongelma, koska suunnittelijoilla ei ole päivitettyjä lähtötietoja käytössään. Toinen merkittävä ongelma on siirrettävän suunnittelutiedon huono hyödynnettävyys. Monia suunnitelmia voidaan käyttää ainoastaan mallina ja esimerkiksi arkkitehdin geometria joudutaan mallintamaan rakennesuunnittelijan toimesta uudelleen.

Algoritmiavusteista suunnittelua voidaan pitää osana tietokoneavusteisen suunnittelun kolmatta sukupolvea (Aish 2013). Algoritmiavusteisessa suunnittelussa suunnittelija ei mallinna rakenteita perinteisin menetelmin vaan luo algoritmien avulla säännöt, joiden mukaan ohjelma osaa mallintaa rakenteet automatisoidusti. Rakennusalalla algoritmiavusteinen suunnittelu tapahtuu pääasiassa jonkin visuaalisen ohjelmointialustan, kuten esimerkiksi Grasshopperin, avulla. Suunnittelussa keskitytään erityisesti erilaisten suunnittelutietojen älykkääseen yhdistelyyn ja toiminnallisen tietoverkon luomiseen (Boeykens 2012). Algoritmiavusteinen suunnitteluprosessi koostuukin tyypillisesti monista perinteisistä suunnittelumalleista ja tietorakenteista sekä niitä yhdistävistä rajapinnoista (Van der Heijden 2015). Algoritmit luovat, muokkaavat ja linkittävät eri lähteistä

saatavaa tietoa siten, että koko suunnitteluprosessia voidaan pitää yhtenäisenä, muutoksiin automaattisesti reagoivana systeeminä.

Tässä tutkimuksessa algoritmiavusteista rakennesuunnitteluprosessia tutkitaan fiktiivisen suunnittelukohteen avulla. Suunnittelussa hyödynnetään perinteisiä suunnitteluohjelmistoja, kuten Excel, Mathcad, RFEM ja Tekla Structures. Suunnittelutieto pyritään luomaan ja linkittämään toisiinsa täysin algoritmipohjaisesti. Suunnitteluprosessin keskiössä ovat suunnittelutiedon luomiseen ja muokkaamiseen käytettävät algoritmimallit. Jotta algoritmimallien käytettävyys ja hyödynnettävyys voidaan maksimoida, tulee niiden luomisessa noudattaa hyvää mallinnustapaa (Magnusson et al. 2017; Zboinska 2015). Algoritmeja luotaessa on kiinnitettävä erityistä huomiota tiedon jaotteluun, ryhmittelyyn, nimeämiseen ja numerointiin. Toinen suunnitteluprosessin toimivuuden kannalta merkittävä asia on tiedonsiirto. Algoritmimallit linkittyvät toisiinsa pilvipalvelun kautta tapahtuvan reaaliaikaisen tiedonsiirron avulla. Tämä toteutetaan Speckle-lisäosaa hyödyntämällä. Lisäksi algoritmit linkittyvät perinteisiin suunnitteluohjelmistoihin niihin tarkoitettujen Grasshopper-lisäosien avulla.

Algoritmiavusteisen suunnitteluprosessin koetaan toimivan pääpiirteissään hyvin. Erityisesti prosessissa tapahtuvan algoritmiavusteisen suunnittelun piirteet vastaavat hyvin siihen kirjallisuudessa liitettäviä ominaisuuksia. Tanska & Österlund (2014) esittävät, että algoritminen suunnittelu mahdollistaa useiden erilaisten vaihtoehtojen nopean vertailun ja nopeuttaa näin iteratiivista suunnitteluprosessia. Tähän samaan lopputulokseen päästään myös tässä tutkimuksessa. Algoritmien parametrisen luonteen ja algoritmien ja suunnittelumallien älykkään integraation ansiosta tietynlaisten muutosten tekeminen on hyvin paljon nopeampaa perinteisiin menetelmiin verrattuna. Erityisesti haastavaan geometriaan kohdistuvat muutokset ovat hyvin helppoja suorittaa, kun perinteisin menetelmin ne olisivat työläitä ja aikaa vieviä. Koska esimerkiksi laskenta- ja rakennemalli voidaan linkittää algoritmien avulla toisiinsa, vältytään päällekkäiseltä mallinnustyöltä ja näin ollen suunnitteluprosessi nopeutuu.

Algoritmiavusteisessa suunnitteluprosessissa suunnittelutiedon luominen, muokkaus ja analysointi pyritään suorittamaan visuaalisessa ohjelmointialustassa. Algoritmien sisältämä suunnittelutieto on yksinkertaisessa ja universaalissa muodossa, minkä ansiosta sen linkittäminen erilaisiin ohjelmiin on helppoa. Lisäksi tiedon prosessointi on selvästi nopeampaa perinteisiin suunnittelumalleihin verrattuna. Algoritmisessa prosessissa voidaan hyödyntää paremmin tietokoneen suurta laskentatehoa ja parantaa näin suunnittelun tehokkuutta. Algoritmien avulla voidaan lisätä myös suunnittelun tarkkuutta. Suunnittelussa ei tarvitse tehdä perinteiselle suunnittelulle tyypillisiä yksinkertaistuksia, kun algoritmit pystyvät ratkaisemaan myös työläät tehtävät täsmällisesti ja nopeasti. Algoritmisten menetelmien ansiosta suunnittelussa ei tarvitse myöskään tukeutua niin vahvasti perinteisiin suunnitteluohjelmistoihin ja niiden tarjoamiin työkaluihin. Esimerkiksi tietomallinnuksesta saadaan huomattavasti joustavampaa, kun liitososien välisiä korrelaatioita

voidaan määrittää algoritmien avulla. Tällöin ei ole välttämätöntä käyttää mallinnusohjelman omia, joskus hyvin rajoittuneita liitoskomponentteja.

Vaikka algoritmiavusteinen suunnittelu koetaan rakennesuunnittelun kannalta hyvin hyödylliseksi menetelmäksi, kohdataan sen käytössä myös heikkouksia ja puutteita. Haasteena on ennen kaikkea tunnistaa ne tilanteet ja tehtävät, joissa algoritmisista menetelmistä voidaan saada lisäarvoa. Usein algoritmeja käytetäänkin vain osana suunnitteluprosessia. Tämä aiheuttaa kuitenkin haasteita erityisesti algoritmisen ja perinteisen suunnittelun rajapinnassa. Jos algoritmisesti ja perinteisin menetelmin tuotettuja rakenteita luodaan samaan malliin, on suunnittelijoiden oltava erityisen tarkkana. Yleisesti ottaen algoritminen tiedonsiirto on suotavaa lopettaa prosessissa ennen kuin mallia aletaan muokata manuaalisesti. Toinen algoritmiavusteisen suunnittelun haaste on mahdollisten muutosten arvioiminen suunnittelun alkuvaiheessa. Jotta algoritmit toimisivat hyvin, tulee niitä luotaessa ottaa huomioon kaikki mahdolliset rakenteisiin kohdistuvat muutokset. Tämä on kuitenkin erityisesti suurien suunnittelukokonaisuuksien kohdalla hyvin vaikeaa. Kolmas suunnitteluprosessissa kohdattava ongelma on reaaliaikaisen tiedonsiirron puutteet. Erityisesti suunnittelukokonaisuuksien kasvaessa ja tietomäärän lisääntyessä reaaliaikaisessa pilvipohjaisessa tiedonsiirrossa kohdataan ongelmia.

Koska suunniteltavat rakennukset eroavat aina jonkin verran toisistaan, on suunnittelun täysi automatisointi hyvin haastavaa. Joissakin yhteyksissä visuaalista ohjelmointia hyödyntävän algoritmiavusteisen suunnittelun odotetaan olevan vain ohimenevä vaihe ennen täysin tekstimuotoista ohjelmointia. On algoritmien muodostamistapa sitten kumpi tahansa, voidaan algoritmiavusteisella suunnittelulla odottaa olevan merkittävä rooli rakennesuunnittelun tulevaisuudessa. Algoritmiavusteinen suunnittelu soveltuu myös muihin rakennusalan tehtäväkenttiin ja sen uskotaan tuovan monia mahdollisuuksia laajamittaisesti koko rakennusalalle. Algoritmiset menetelmät mahdollistavat täysin uudenlaisen yhteistyön, jonka ansiosta koko rakennushankkeen toiminta voidaan saada tehokkaammaksi. Uudet menetelmät mahdollistavat näin entistä tarkemman, tehokkaamman ja edullisemman suunnittelun ja rakentamisen.

LÄHTEET

Aish, R. (2013). First Build Your Tools, in: Peters, B. & Peters, T. (ed.), Inside Smartgeometry: Expanding the Architectural Possibilities of Computational Design, John Wiley & Sons Ltd, Chichester, West Sussex, United Kingdom, pp. 36–49.

Avern-Taplin, A., Birmingham, N., Buxton, S. & Riddle, C. (2016). Elizabeth Quay Pedestrian and Cyclist Bridge, The Arup Journal, Vol. 51(1), pp. 22–31.

Bourque, P. & Fairley, R.E. (2014). Guide to the Software Engineering Body of Knowledge, Version 3.0, IEEE Computer Society. Saatavissa: www.swebok.org

Brown, N., de Oliveira, J.I.F., Ochsendorf, J. & Mueller, C. (2016). Early-stage integration of architectural and structural performance in a parametric multi-objective design tool, Proceedings of the Third International Conference on Structures and Architecture, Guimarães, Portugal, 27–29 July, 2016, CRC Press, London, UK, pp. 1103–1111.

Grasshopper Primer (2014). Akos, G. & Parsons, R., Mode Lab, 143 p. Saatavissa: https://aae280.files.wordpress.com/2014/10/mode-lab-grasshopper-primer-third-edition.pdf

Bentley Communities, Generative Components, verkkosivu. Saatavissa (viitattu 30.11.2017): https://communities.bentley.com/products/products_generativecomponents/f/generativecomponents---forum/143682/yes-no-or-true-false-node

Boeykens, S. (2012). Bridging Building Information Modeling and Parametric Design, eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2012, CRC Press, London, United Kingdom, pp. 453–459.

Davis, D. (2013). Modelled on Software Engineering: Flexible Parametric Models in the Practice of Architecture, dissertation, RMIT University, 234 p. Saatavissa: http://www.danieldavis.com/papers/danieldavis_thesis.pdf

Davis, D., Burry, J. & Burry, M. (2011). Understanding Visual Scripts: Improving Collaboration through Modular Programming, International Journal of Architectural Computing, Vol. 9(4), pp. 361–375.

Dynamo Forum, verkkosivu. Saatavissa (viitattu 30.11.2017): https://forum.dynamobim.com/t/add-shared-parameters-failing-if-parameter-already-exists/16858

Grasshopper-Tekla Live Link, Tekla Structures Support, verkkosivu. Saatavissa (viitattu 20.6.2018): https://teklastructures.support.tekla.com/not-version-specific/en/ext_grasshopperteklalink

Haahtela, Y. & Kiiras, J. (2014). Talonrakennuksen kustannustieto 2014, Haahtela-kehitys, Helsinki, 390 s.

Harding (2014). Meta-Parametric Design. Developing a Computational Approach for Early Stage Collaborative Practice, dissertation, University of Bath, 283 p.

IEC 60050-351:2013, 351-42-27 (2013). Kansainvälinen elektrotekninen sanasto, Algoritmi, IEC Technical Committee 1.

Janssen, P. & Chen, K. W. (2011). Visual Dataflow Modelling: A Comparison of Three Systems, Proceedings of the 14th International conference on Computer Aided Architectural Design, Les Éditions de l'Université de Liège, Liège, Belgium, pp. 801–816.

Junnonen, J.-M. & Kärnä, S. (2015). Suunnitelmien virheettömyydellä tehokkaampaan rakentamiseen. Tiivistelmä RALA-projektipalautetiedon suunnittelijapalautteen analyysista, 17 s. Saatavissa: http://www.rala.fi/tiedostot/Suunnittelijapalaute_tulostiivistelma.pdf

Kerosuo, H., Mäki, T., Codinhoto, R., Koskela, L. & Miettinen, R. (2012). In time at last: Adaption of Last Planner tools for the design phase of a building project, 20th Annual Conference of the International Group for Lean Construction, San Diego, CA, United States, pp. 1031–1041.

Koskenvesa, A. (2010). Rakennustyön tuottavuus 1975-2010. in Junttila, A., Koskenvesa, A., Heloma, T. & Laine, S. (ed.), Rakentajain Kalenteri 2011, Rakennustietosäätiö RTS, Helsinki, s. 138–146.

Lassila, R. (2016). Tietomallintaminen ja LEAN-työskentely rakennushankkeen suunnittelunohjauksen apuvälineenä, diplomityö, Tampereen teknillinen yliopisto, Arkkitehtuurin laitos, Tampere, 149 s. Saatavissa: http://URN.fi/URN:NBN:fi:tty-201609084495.

Lindholm, M. (2009). Kustannushallinta rakennushankkeessa, Suomen Rakennusmedia Oy, Helsinki, 56 s.

Maankäyttö- ja rakennuslaki (1999). 17.1.2014/41. Saatavissa (viitattu 14.9.2018): https://www.finlex.fi/fi/laki/ajantasa/1999/19990132.

Macaw, App, verkkosivu. Saatavissa (viitattu 20.6.2018): http://www.food4rhino.com/app/macaw.

Magnusson, F., Runberger, J., Zboinska M.A. & Ondejcik, V. (2017). Morphology & Development - knowledge management in architectural design computation practice, Proceedings of the 35th eCAADe Conference, Rome, Italy, September 20–22, 2017, pp. 683–690.

MetaHopper, App, verkkosivu. Saatavissa (viitattu 20.6.2018): http://www.food4rhino.com/app/metahopper.

Naumanen, S. (2015). Hyvän suunnittelun vaikuttavuus rakennushankkeen onnistumiseen, diplomityö, Tampereen teknillinen yliopisto, Rakennustekniikan laitos, Tampere, 91 s. Saatavissa: https://dspace.cc.tut.fi/dpub/bitstream/handle/123456789/23547/naumanen.pdf?sequence=1&isAllowed=y

Oxman, R. (2006). Theory and design in the first digital age, Design Studies, Vol. 27(3), pp. 229–265.

Pro IT -tutkimusraportti (2004). Tuotemallinnus rakennesuunnittelussa. Perusteet ja ohjeita. Rakennusteollisuus RT ry. 55 s. Saatavissa: http://virtual.vtt.fi/virtual/proj6/proit/

Rhinoceros 3D, NURBS, verkkosivu. Saatavissa (viitattu 13.12.2017): https://www.rhino3d.com/nurbs.

RIL 229-1-2013 (2013). Rakennesuunnittelun asiakirjaohje. Tekstiosa, Suomen Rakennusinsinöörien Liitto RIL ry, 173 s.

RT 10-10992 (2010). Tietomallinnettava rakennushanke. Ohjeita rakennuttajalle. Rakennustieto Oy. 13 s.

RT 10-11068 (2012). Yleiset tietomallivaatimukset 2012. Osa 3. Arkkitehtisuunnittelu. Rakennustieto Oy. 17 s.

RT 10-11070 (2012). Yleiset tietomallivaatimukset 2012. Osa 5. Rakennesuunnittelu. Rakennustieto Oy. 18 s.

RT 10-11080 (2012). Yleiset tietomallivaatimukset 2012. Esittely. Rakennustieto Oy. 4 s.

RT 10-11222 (2016). Talonrakennushankkeen kulku. Rakennushankkeen osapuolet. Rakennustieto Oy. 6 s.

RT 10-11224 (2016). Talonrakennushankkeen kulku. Rakennushankkeen vaiheet ja osittelu. Rakennustieto Oy. 4 s.

RT 10-11226 (2016). Talonrakennushankkeen kulku. Kustannusten muodostuminen ja ohjaus. Rakennustieto Oy. 5 s.

RT 10-11128 (2013). Rakennesuunnittelun tehtäväluettelo RAK12. Rakennustieto Oy. 28 s.

Scott Tallon Walker Architects, Aviva Stadium, verkkosivu. Saatavissa (viitattu 8.12.2017): http://www.stwarchitects.com/data/sketchbook/publications/brochures/Green%20Building%20Award_Aviva_Winner.pdf

Speckle Works, verkkosivu. Saatavissa (viitattu 20.6.2018): https://speckle.works/.

Sulankivi, K., Lakka, A. & Luedke, M. (2002). Projektin hallinta sähköisen tiedonsiirron ympäristössä, VTT Tiedotteita 469, VTT, Espoo, 162 s. Saatavissa: http://www.vtt.fi/inf/pdf/publications/2002/P469.pdf

Tanska, T. & Österlund, T. (2014). Algoritmit puurakenteissa: menetelmät, mahdollisuudet ja tuotanto, B 32, 1st ed. DigiWoodLab, Oulun yliopisto, Arkkitehtuurin tiedekunta.

Van der Heijden, R., Levelle, E. & Riese, M. (2015). Parametric Building Information Generation for Design and Construction, Proceedings of the 35th Annual Conference of the Association for Computer Aided Design in Architecture, Cincinnati, United States, October 19–25, 2015, pp. 417–429.

Woodbury, R. (2010). Elements of Parametric Design, Routledge, New York, United States, 300 p.

Ympäristöministeriön asetus kantavista rakenteista (2014). 477/2014. Saatavissa (viitattu 14.9.2018): https://www.finlex.fi/fi/laki/alkup/2014/20140477.

Zboinska M.A. (2015). Boosting the Efficiency of Architectural Visual Scripts, in: Thomsen M., Tamke M., Gengnagel C., Faircloth B. & Scheurer F. (ed.), Modelling Behaviour. Springer International Publishing, Switzerland, 2015, pp. 479–490.